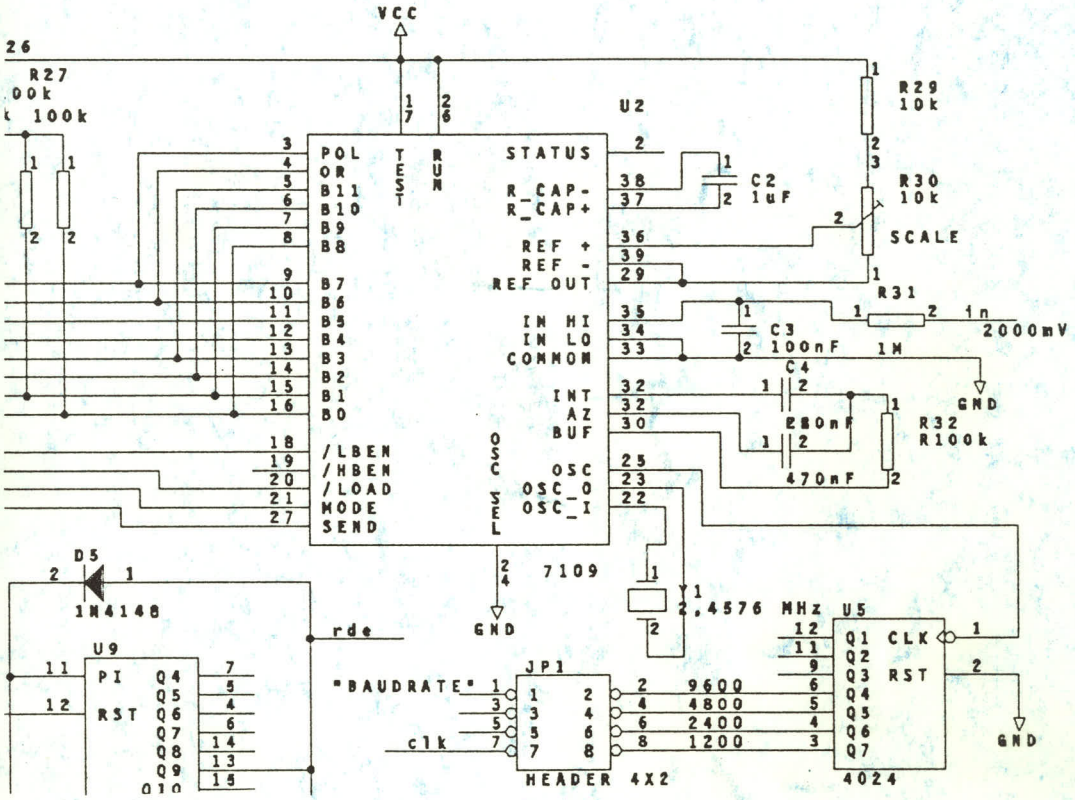
De µP Kenner

Zeventiende jaargang nr. 2
April 1993
81



In dit nummer o.a.:

C: Cursus!
Intel's vijfde
Thermometer op RS-232
Geheugenbeheer bij de 680X0
TEXtverwerken kan ook anders

Inhoudsopgave

De μP Kenner

Nummer 81, april 1993
Verschijnt 5 maal per jaar
Oplage: 250 stuks
Druk: FEBO Offset, Enschede

De redactie:

Gert van Opbroek
Geert Stappers
Nico de Vries

Eindredactie:

Gert van Opbroek

Vormgeving:

Joost Voorhaar
Nico de Vries

Redactieadres:

p/a Gert van Opbroek
Bateweg 60
2481 AN Woubrugge

De μP Kenner nummer 82 verschijnt op 21 augustus 1993.

Kopijsluitingsdatum voor nummer 82 is vastgesteld op 7 augustus 1993.

Vereniging

Uitnodiging clubbijeenkomst 5
Voortgang KGN68k deel 12 40
Van de bestuurstafel 53

Algemeen

Redactioneel 4
Stoeien met PostScript, deel 3 25
Bus-oorlogen 52
Opmerkingen bij de poorten 52
Nieuwtjes en andere wetenswaardigheden 53

Software/Talen

Een echt document-opmaak-systeem: TEX en LATEX ... 15
Nogmaals Programmeertalen 27
Eerste les "C" 29

Hardware

RS-232 Thermo Interface 6
Intel is de tel kwijt... 13
Teletekst (deel 3) 38

DOS65

Voortgang DOS65 50

Systemen

De Motorola 68030; het hart van KGN68k (deel 5) 41

De μP Kenner is het huisorgaan van de KIM gebruikersclub Nederland en wordt bij verschijnen gratis toegezonden aan alle leden van de club. De μP Kenner verschijnt vijf maal per jaar, in principe op de derde zaterdag van de maanden februari, april, augustus, oktober en december.

Kopij voor het blad dient bij voorkeur van de leden afkomstig te zijn. Deze kopij kan op papier, maar liever in machine-leesbare vorm opgestuurd worden aan het redactieadres. Kopij kan ook op het Bulletin Board van de vereniging gepost worden in de redactie area.
Nadere informatie kan bij het redactieadres of via het bulletin board opgevraagd worden.

De redactie houdt zich het recht voor kopij zonder voorafgaand bericht niet of slechts gedeeltelijk te plaatsen of te wijzigen. Geplaatste artikelen blijven het eigendom van de auteur en mogen niet zonder diens voorafgaande schriftelijke toestemming door derden gepubliceerd worden, in welke vorm dan ook.
De redactie noch het bestuur kan verantwoordelijk gesteld worden voor toepassing(en) van de geplaatste kopij.

Redactioneel

Hoe bestaat het, hier is al weer een μP Kenner. Inderdaad het heeft weer het nodige bloed zweet en tranen gekost maar het is toch gelukt de μP Kenner voor de mei-bijeenkomst bij de leden te hebben. Verder is het ook een behoorlijk goed gevuld nummer (misschien mag je wel zeggen: uitzonderlijk dik). Dat komt doordat er enkele auteurs zijn die in dit nummer voor het eerst een bijdrage leveren (en wat voor bijdrage!). Ik stel dat uiteraard zeer op prijs en wil mensen die nog twijfelen nogmaals oproepen eens een keer iets voor het blad te schrijven tenslotte is de μP Kenner voor de leden en bij voorkeur ook gemaakt door de leden.

Mijn oproep voor een 'C' cursus in het blad heeft succes gehad. U vindt in deze uitgave (en de vijf volgende zijn mij toegezegd) een bijdrage van Hans van Boheemen (inderdaad een nieuwe auteur!). Het is de bedoeling dat hij een cursus 'C' gaat schrijven, te beginnen op het zogenaamde nul-niveau zodat mensen die nu nog niet in 'C' kunnen programmeren dat kunnen leren. Ik stel mij daar zeer veel van voor want ik heb begrepen dat een behoorlijk deel van de leden graag 'C' wil gaan gebruiken. Verder wordt de software voor KGN68k ook bijna uitsluitend in 'C' ontwikkeld zodat het ook voor de groep mensen die deze ontwikkeling volgt van belang is.

Wat er ook in het blad staat is de beschrijving van de software voor de Teletekst Decoder. Het is uiteraard ondoenlijk de complete programma's in het blad te plaatsen maar die kunnen via The Ultimate of op bijeenkomsten gekopieerd worden. Ik heb met Herman Hek gesproken en hij vertelde mij over wat voor plannen er nog bestaan met dit project, wel dat is ook niet misselijk; ik ben zeer benieuwd.

Het derde en laatste onderdeel dat ik even aan wil stippen (zonder daarmee de andere auteurs te kort te doen) is het artikel van Phons Bloemen. Zoals de meeste mensen wel weten, wordt ons blad opgemaakt met behulp van een professioneel pakket waarvan ik de naam niet wil noemen. Phons Bloemen beschrijft een pakket dat, voor zover ik weet, Public Domain (of zoiets; in ieder geval goed beschikbaar) is. Om de kracht van dit pakket aan te tonen heeft hij zijn artikel zelf opgemaakt, uitgedraaid en op papier aangeleverd. U kunt dus zelf constateren of dit pakket zich kan meten met de professionele pakketten zoals Ventura of Pagemaker. Voor de mensen die nog niet overtuigd zijn van de mogelijkheden van TeX (spreek uit "tech") zal hij op de meibijeenkomst ook nog een voordracht over (en waarschijnlijk een demonstratie van) dit pakket geven.

Tja, en dan moet ik het toch nog even hebben over de continuïteit van de KGN en de μP Kenner. Zoals bekend, heeft de club momenteel nog slechts vier bestuursleden. Verder hebben al deze bestuursleden een drukke baan en een sociaal leven buiten de KGN (o.a. in twee gevallen een gezin met kleine kinderen). Dat betekent dat er momenteel zo nu en dan bestuurstaken gewoon blijven liggen omdat de bestuursleden anders overbelast raken. En valt er dan ook nog een bestuurslid tijdelijk uit, dan wordt het helemaal moeilijk. Wat ik hiermee wil zeggen, is dat het huidige bestuur graag zou zien dat minstens een deel van de drie vacatures die we in het bestuur hebben opgevuld worden. Aangezien we op elke ledenvergadering een bestuursverkiezing kunnen houden is het dus nu het moment voor de mensen die in november nog even de kat uit de boom hebben gekeken. Meld je aan zodat je in mei verkozen kunt worden waarna we aan het begin van het volgende computer-seizoen weer een slagvaardig bestuur hebben.

Hetzelfde verhaal geldt ook voor de redactie. In de eerste plaats kan ik uiteraard weer veel kopij gebruiken. Het volgende nummer komt in augustus uit zodat u voor deze ene keer drie maanden de tijd hebt nu eindelijk eens dat artikel te schrijven dat u al drie jaar wilde schrijven. O, u hebt kinderen? Dat is toch alleen maar een voordeel! Dan schrijft u toch een testrapport over de spelletjes die uw kinderen altijd op de computer spelen. Kan Pa of Ma de computer ook eens gebruiken. Verder zoek ik naast incidentele auteurs ook mensen die een vaste rubriek in het blad verzorgen. Neem als voorbeeld de rubriek ShareWare van Joost Voorhaar of de nieuwe 'C'-cursus. Het lijkt me namelijk verstandig de redactie wat meer zwaarte te geven zodat pakweg de helft tot driekwart van het blad met vaste onderwerpen gevuld kan gaan worden.

Tenslotte wil ik aan allen die meegewerkt hebben aan deze of één van de vorige uitgaven van de μP Kenner nog even het compliment doorgeven dat de hoofdredacteur van Elrad (een nieuw computer-/elektronika-blad) ons gaf: "Het ziet er goed uit".

Tot ziens op de bijeenkomst op 15 mei en anders een prettige vakantie en veel plezier met uw computerhobby. Houdt u ook rekening met de datum 22 juni? Dat is de verjaardag van de KGN (zie de laatste pagina). Ik wil in het augustusnummer op dit derde lustrum van onze club terugkomen; tot dan!

Gert van Opbroek.

Uitnodiging clubbijeenkomst

Datum: 15 mei 1993
Locatie: PMWZ-Oost
Bouwstraat 55
3575 Utrecht
Thema: Jaarverslag 1992
Desktop Publishing
Entree: Leden gratis, niet-leden fl. 10,--

Routebeschrijving

Bereikbaarheid: Er is slechts weinig parkeergelegenheid in de buurt. Als je dus geen apparatuur mee te slepen hebt, raden wij je sterk aan per openbaar vervoer te komen.

Openbaar vervoer:

Vanaf het centraal station kun je verder met de bus: lijn 4 (uitstappen bij de Biltstraat) of lijn 11 (uitstappen bij de Kruisstraat). Als je lijn 4 neemt (en uitstapt op de Biltstraat) loop je de Obrechtstraat in. De tweede zijstraat aan de rechterhand is dan de Bouwstraat. Neem je lijn 11, dan loop je de Kruisstraat uit richting het noorden. Dan kom je ook op de Biltstraat uit. Daar ga je dan linksaf (de Biltstraat op); de tweede rechts is dan vervolgens de Obrechtstraat. Zie verder bij lijn 4.

Per auto:

Als je niet via de A27 vanuit Hilversum komt, benader Untrecht dan via de A27 vanuit het zuiden. Rij door tot de afslag Veemarkt (Utrecht Oost). Komt u wel van Hilversum via de A27, dan is het volgens mij niet mogelijk bij de genoemde afslag de A27 te verlaten. Rij door tot aan het knooppunt Rijnsweerd en keer daar om.

Na de afslag Veemarkt ga je richting Utrecht centrum, de Biltse Rading op (linksaf). Op het eerste plein sla je linksaf de Sartreweg in. Doorrijden tot aan de Berekuil. Daar ga je rechtsaf de Biltstraat op. Dan rechtsaf de Obrechtstraat in. Pas op: als je op de Witte Vrouwensingel uitkomt ben je al te ver! Steek de Frederikastraat over; de volgende is dan de Bouwstraat.

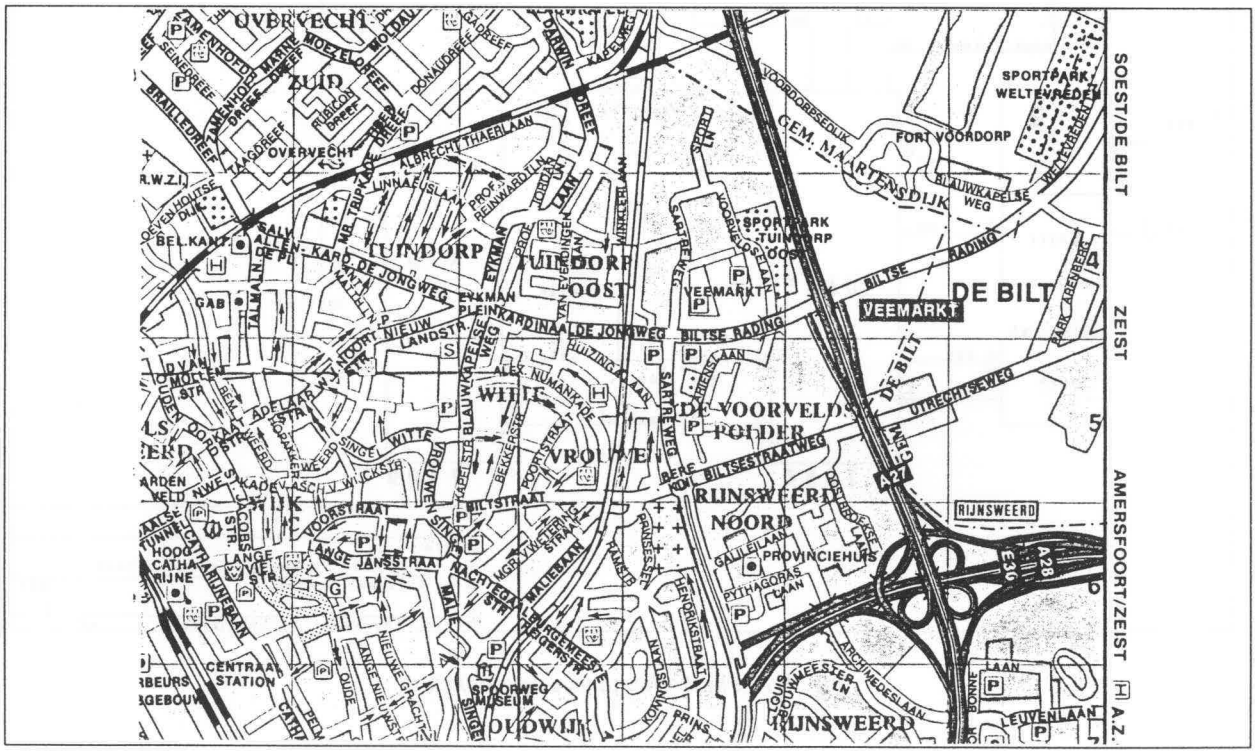
Programma:

9:30 Zaal open met koffie
10:00 Opening
10:05 Algemene ledenvergadering met als agenda:
- Opening ledenvergadering/vaststellen agenda
- Verslag ledenvergadering d.d. 28-11-1992
- Behandeling jaarverslag 1992
- Rondvraag

11:00 Voordracht van Phons Bloemen over TeX
12:30 Lunchpauze
13:30 Forum en Markt

Aansluitend het informele gedeelte met de mogelijkheid om andermans systemen te bewonderen en Public Domain software uit te wisselen. U en uw systeem zijn uiteraard van harte welkom.

17:00 Sluiting.



RS-232 Thermo Interface

Inleiding

Op zoek naar steeds meer mogelijkheden, heb ik sinds enkele maanden een nieuwe meter voor het sturen van het bakproces van een pottenbakkersoven. Door gebruik te maken van de RS-232 poort, is de meter op elke computer aan te sluiten, zonder dat er iets aan de hardware van de computer veranderd hoeft te worden. Bovendien kan de schakeling eenvoudig aangepast worden aan een andere toepassing.

De schakeling is opgebouwd rond een ouwe getrouwe, de A/D convertor 7109 van Intersil, aangevuld met een UART en nog enkele componenten. Vijf omzettingen per seconde is ruim voldoende voor het controleren van het trage bakproces. De nauwkeurigheid, met een omzetting naar 12 bits, is veel belangrijker omdat we temperaturen tot 1300 C en meer willen meten.

In de schakeling zijn enkele "vreemde zaken" die eigen zijn aan dit ontwerp. Het zijn enkele eigen ontworpen veiligheden, want uiteindelijk willen we geen scherven uit de oven halen.

Het blokschema

In figuur 1 staat het blokschema afgebeeld. De A/D converter staat continu te meten. De interne referentiebron wordt teruggekoppeld en ingesteld op 1024 mV. Hierdoor wordt de volle schaal 2048 mV. De resolutie van 12 bits komt overeen met 4096 zodat het laagste bit een waarde krijgt van 0.5 mV.

Door het kiezen voor een kristal van 2.4576 MHz werkt de A/D converter iets trager, maar kan de gebufferde oscillator uitgang dienen als basis voor de baudrate van de UART.

Met de verbindingen RxD en TxD is de schakeling met de computer verbonden. Er wordt geen gebruik gemaakt van Handshake-signalen. De schakeling wordt wakker geschud door het ontvangen van een byte. De UART geeft zo aan de A/D convertor de opdracht om het resultaat van de laatste omzetting door te geven. De A/D converter vult nu twee maal

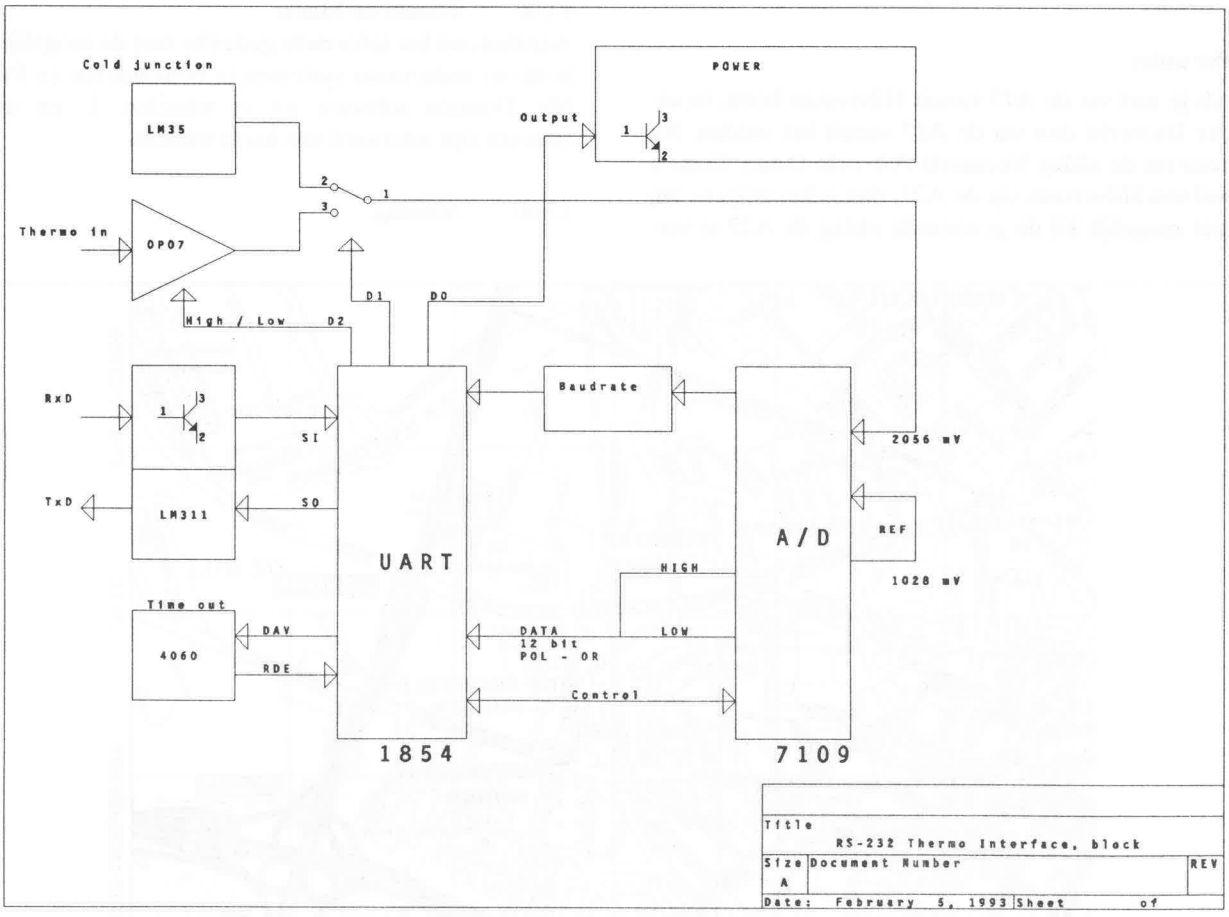


Fig. 1: Het blokschema van de schakeling

de transmitterbuffer van de UART en de twee bytes worden doorgezonden naar de computer. Dus één byte naar binnen en twee bytes terug, eerst het high-byte met een polariteitsaanduiding en een overrun, daarna het low-byte.

Tijdens de handshake-mode (MODE ingang is hoog) van de A/D converter zijn een aantal ingangen tot een uitgang veranderd: /CE + /LOAD, /LBEN en /HBEN zijn uitgangen en SEND is een ingang.

Bij het wakker schudden van de schakeling gebeurt er nog iets. Het DAV-signaal van de UART reset de 4060. Deze time-out teller geeft de uitgang van de UART vrij waardoor het ontvangen byte de andere delen van de schakeling kan besturen. Zo wordt de uitgang, waarop de pottenbakkersoven is aangesloten, d.m.v. bit D0, hardware-matig uitgeschakeld als er iets mocht fout gaan met de computer of de communicatie. Bit D1 van de binnenkomende byte bepaalt wat er gemeten wordt: ofwel de spanning van het thermokoppel, ofwel de spanning van de LM35.

Ervaren pottenpakkers, sorry, pottenbakkers, merken echter wel een verschil tussen een zomer- en een winterbak.

De twee laatste blokken zijn de niveau-aanpassingen van het RS-232 signaal en natuurlijk de voeding van de gehele schakeling.

Thermokoppels

In de industrie worden thermokoppels zeer veel gebruikt. Het zijn zeer goedkope en vooral betrouwbare temperatuurvoelers. Als elektriciteit warmte kan opwekken, dan moet het omgekeerde ook kunnen (wet van behoud van energie). Dat is ook zo. Thermokoppels zijn eigenlijk thermo-elementen, een stroombron die ontstaat doordat twee verschillende metalen (of legeringen) met elkaar verbonden worden. Koper heeft meer vrije elektronen dan constantaan. Bij contact gaan de vrije elektronen over van koper, dat positief wordt, naar het constantaan en zo ontstaat een contactspanning. De grootte hangt af van de aard van de metalen. In een gesloten kring (met twee overgangen) zijn de contactspanningen even groot en tegengesteld, op voorwaarde dat de overgangen even warm zijn. Dit wordt dan ook het principe van de meting: het ene contactoppervlak wordt verwarmd (door de warmte van de oven e.d.) waardoor de elektronen overgang versterkt wordt, en dus ook de contactspanning. Het andere contactoppervlak houden we constant. Door in de kring een galvanometer op te nemen kunnen we nu rechtstreeks de temperatuur meten. Eigenlijk meten we de verschiltemperstuur tussen de warme en de koude las, de contactoppervlakten. Willen we zeer nauwkeurig werken, dan moeten we rekening houden met de temperatuur van de koude las en deze optellen bij de verschiltemperatuur van het thermokoppel. Dit is de koudelas compensatie (cold junction).

In de praktijk is er altijd een afstand tussen de plaats waarop we de temperatuur willen meten en de plaats waar we de temperatuur willen aflezen. Tussen het thermokoppel en de meter gebruiken we hiervoor een compensatiekabel. De koude las is nu verplaatst naar het uiteinde van de compensatiekabel. De compensatiekabel wordt meestal wel gebruikt, maar het optellen van de temperatuur van de koude las bij de verschiltemperatuur van het thermokoppel wordt meestal niet gedaan. Ervaren pottenpakkers, sorry, pottenbakkers, merken echter wel een verschil tussen een zomer- en een winterbak (of pak ?): een verschil van enkele tientallen graden op duizend (dus ongeveer 2 %). We moeten eigenlijk al zeer nauwkeurig werken om deze meetfout op de juiste wijze weg te werken.

In deze schakeling meet de LM35 de temperatuur van de koude las. De LM35 is eigenlijk te nauwkeurig voor deze toepassing, maar wel zeer eenvoudig in gebruik doordat er geen afregeling nodig is. De spanningsafgifte (stroom over weerstand) van een thermokoppel is zeer klein, het verloop is niet altijd rechtevenredig en elk thermokoppel heeft zo zijn eigen temperatuursbereik:

koper/constantaan:	27,41 mV bij 500 C
ijzer/constantaan:	46,22 mV bij 800 C
nikkel-chroom/nikkel:	45,16 mV bij 1100 C
platina-rhodium/platina:	14,337 mV bij 1400 C

De eerste twee worden zeer veel in de industrie gebruikt. Wie alleen aardewerk maakt, gebruikt meestal een NiCr/Ni thermokoppel. Het vele duurdere PtRh/Pt thermokoppel heb je echt nodig om regelmatig steengoed of porselein te bakken.

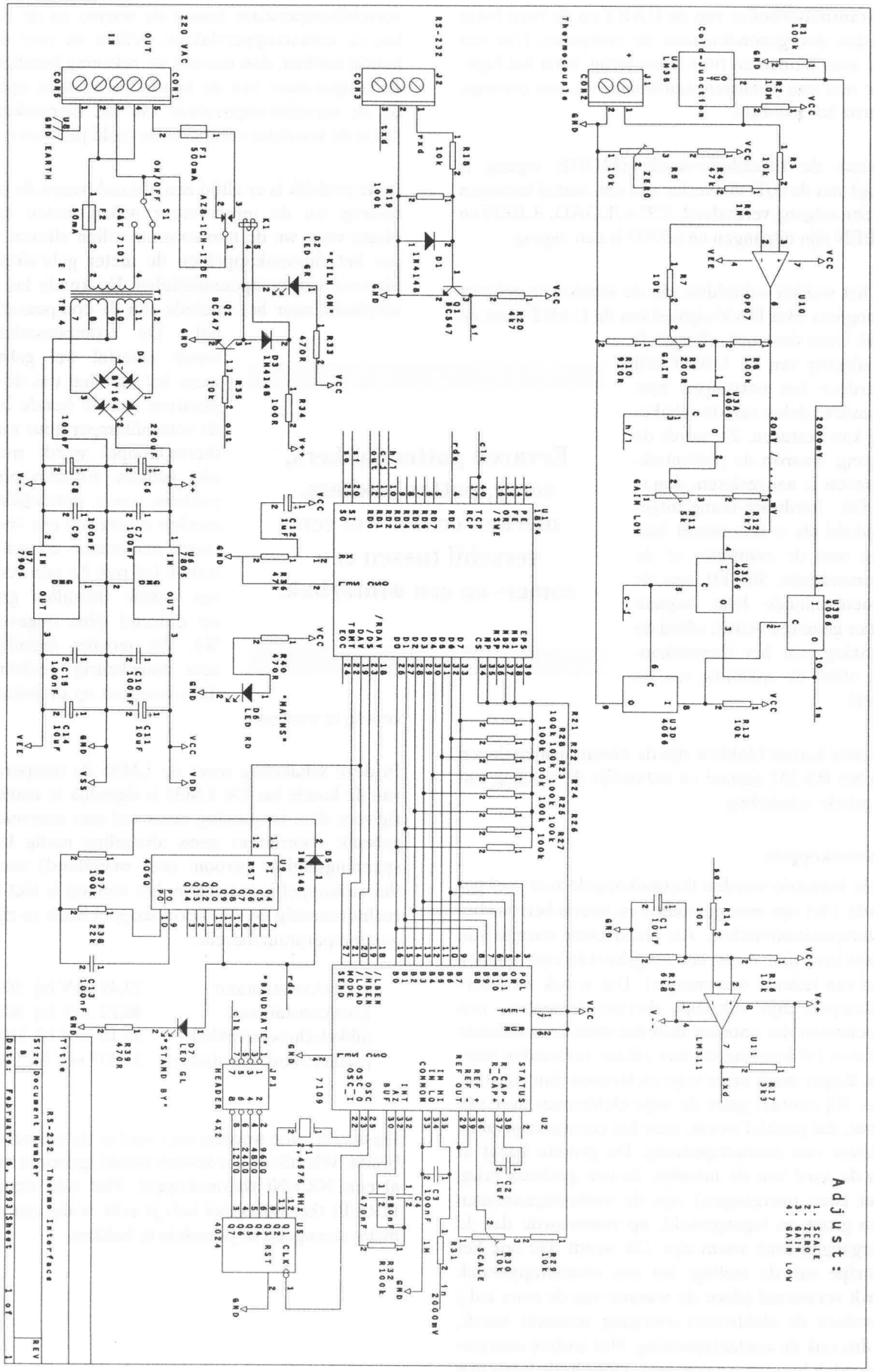


Fig. 2: Het schema van de schakeling

En nu het schema

Waarschijnlijk zijn de verschillende blokken snel terug te vinden. Het kloppende hart, de A/D converter 7109, staat midden-rechts. De waarden van de condensatoren aan INT en AZ (C4, 220nF en C5, 470nF) zijn aangepast aan de clockfrequentie. De waarde van Rint met 100k aan BUF komt overeen met een 1.0 V REF. De clockfrequentie van 2.4576 MHz is beschikbaar op pen 25 en gaat direct naar de deler 4024. Met een jumper kunnen we uit vier snelheden kiezen: van 1200 tot 9600 BAUD. De snelheid van ontvangen en verzenden is gelijk. Bovendien is de UART vast ingesteld op 8N1 (8 bits, geen pariteit en één stopbit). De UART is een CMOS-type en daarom zijn de data-ingangen van pull-up weerstanden voorzien. Met een eenvoudig weerstandje en een condensator wordt bij power-on de UART gereset.

Links onder de A/D conv. zien we de time-out schakeling: de 4060 heeft een eigen RC-oscillator en staat voortdurend te tellen en te delen. Na veel deelwerk zou ook eens Q9 hoog moeten worden, maar tijdens een normaal gebruik wordt dat verhinderd doordat de UART, bij het ontvangen van een byte, met zijn DAV-puls de deler steeds opnieuw reset. Daardoor blijft Q9 laag, wat doorgegeven wordt aan de RDE van de UART, en is de ontvangen byte beschikbaar op de uitgang. Als er geen bytes meer binnenkomen, kan de 4060 eindelijk eens goed doordelen en wordt op een gegeven moment Q9 toch hoog. Dan is het weer gedaan met delen, want D5 maakt de clock-ingang nu continu hoog waardoor de teller stopt. Dit hoge niveau van Q9 gaat ook naar de RDE van de UART, de uitgangen gaan in tri-state waardoor de belangrijkste uitgang, RD0, steeds laag gaat (en de oven uitgeschakeld wordt). Bovendien gaat er een ledje branden "stand-by" om aan te geven dat de schakeling geduldig staat te wachten.

De RS-232 drivers zijn de eenvoud zelve: een transistor tussen RxD en SI en de comparator LM311 tussen SO en TxD.

De ingangsversterker is ook zeer eenvoudig. De zeer stabiele opamp OP07 staat ingesteld op 100 x versterken. Dit is bij te regelen met de trimmer R9 "GAIN". Deze versterking kan omgeschakeld worden naar een lagere waarde, regelbaar met R11 "GAIN LOW", door een weerstand parallel te schakelen m.b.v. de CMOS schakelaar 4066.

Indien het thermokoppel zou breken of slecht contact zou maken, zorgt R2 ervoor dat de ingang hoog getrokken wordt. Daardoor wordt de oven altijd direct uitgeschakeld, want de software meet een zeer hoge oventemperatuur of OR (Over Range). Dan wordt aan de ingang nog een optelling gemaakt met een regelbare offset om de uitgang van de opamp naar nul volt te regelen.

De LM35 lijkt een beetje op een transistor: de "emitter" aan de ground, de "collector" aan de +5 Volt en de "basis" is de uitgang. Door het toevoegen van de weerstand R1 naar -5 Volt kunnen er ook negatieve temperaturen gemeten worden.

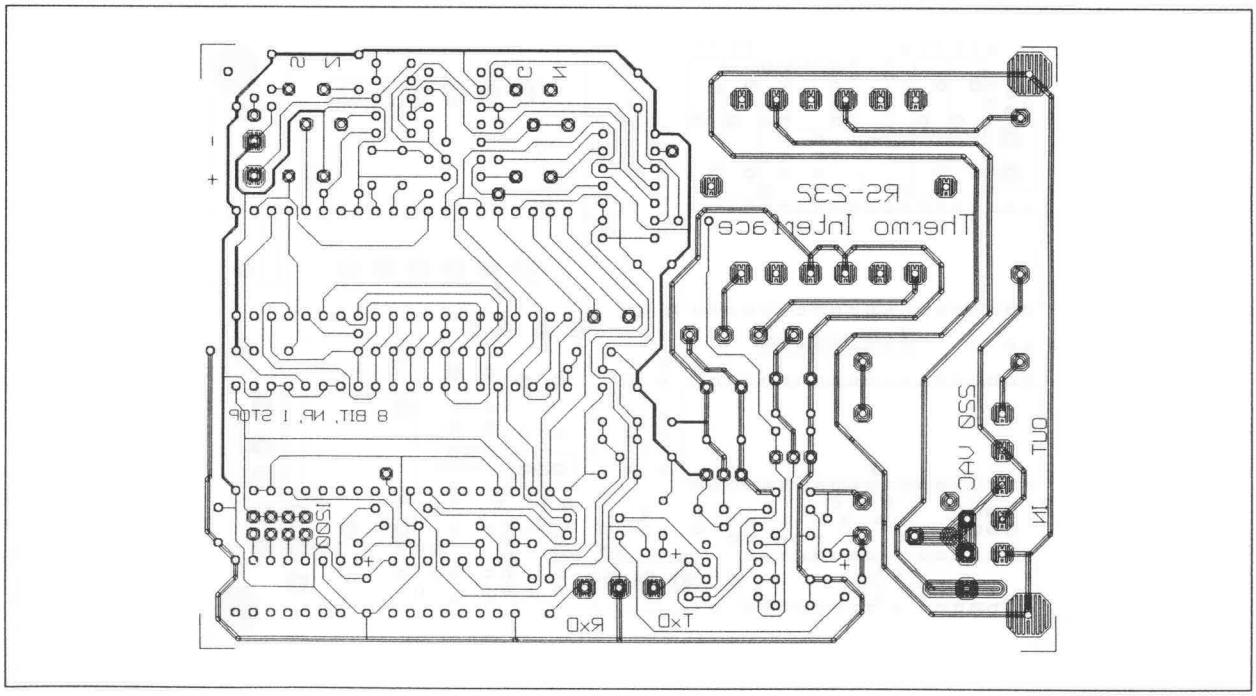


Fig. 3: de print

Drie CMOS schakelaars van de 4066 vormen samen een omschakelaar. U3D inverteert het stuursignaal waardoor ofwel U3C ofwel U3B een spanning doorlaten naar de A/D convertor.

De voeding is met zijn 4 Watt transformator ruim berekend. Een ledje geeft aan dat het apparaat ingeschakeld is. Bij de voeding hoort ook nog de ovensturing. De uitgang RD0 van de UART wordt door een transistor gebufferd en schakelt een kleine relais. Deze schakelt de 220 VAC door naar de vermogenrelais van de oven. Een derde ledje geeft de status van de ovensturing aan.

De print

Alles past op een enkelzijdig printje van ongeveer 8 x 12 cm. Vijf draadbruggen zijn er nog overgebleven nadat de componenten en sporen verschillende keren van plaats zijn verhuisd. Doordat is project is stapjes is ontwikkeld, is de ORCAD-tekening later gemaakt. Aan de ruwe componenten opstelling is dit te zien: de voornaamste onderdelen zijn aangeduid. Wie deze schakeling wil nabouwen, kan van mij alles bekomen: van schema tot lay-out en de verdere datails.

De software

Uit de beschrijvingen weten we al dat de RS-232 communicatie standaard verloopt tegen een snelheid van 1200 baud, 8N1. Daarbij versturen we steeds één byte en direct ontvangen we de laatste conversie in twee bytes.

SEND A BYTE:
- D0: output on/off (1/0)
- D1: read mV from LM35 or thermo (1/0)
- D2: low/high gain (1/0) to read NiCr or PtRh thermo
- D3 ... D7: don't care

LATEST RESULT: (first high byte, then low byte)
HIGH BYTE:
- D7: POL: 1/0 = POS/NEG (!)
- D6: OR: 1 = overranged
- D5 + D4 : always high
- D3 ... D0: high byte of 12 bit result with D3 as most significant bit

LOW BYTE:
- D7 ... D0: low byte of 12 bit result with D0 as lsb

Hoewel de individuele toepassing van de schakeling bij iedereen verschillend zal zijn, moet de software steeds eerst de RS-232 configureren. In het verdere verloop zal de software steeds een geprepareerde byte verzenden (wat wordt mijn volgende meting ?) en daarna het antwoord ontleden om het te verwerken.

Met de assembler en de goede documentatie van DOS-65 heb ik al een aardig programma dat de temperatuur in de oven "gecontroleerd" laat stijgen/dalen op verschillende manieren. Daarbij krijg ik een grafiekje op een klein plottertje met evt. het verbruik.

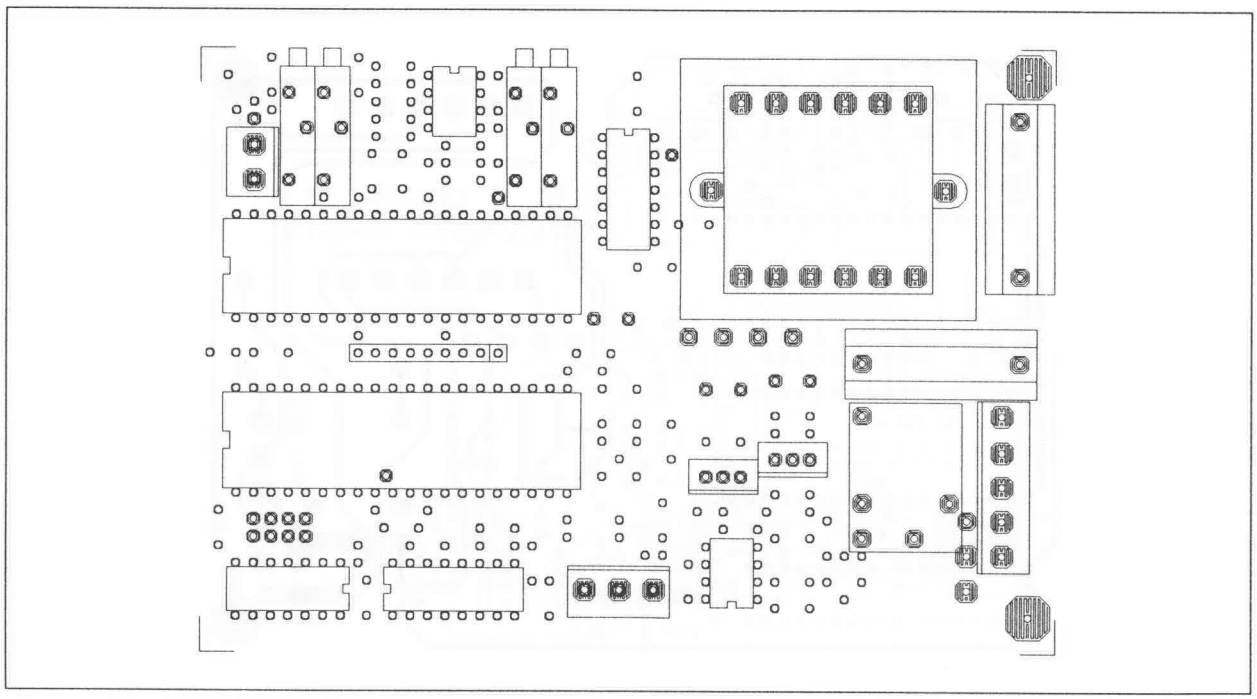


Fig 4: de componentenopstelling

Aantal	Referentie	Waarde
12	R1,R8,R19,R21,R22,R23, R24,R25,R26,R27,R28,R37	100k
1	R2	10M
10	R3,R5,R7,R13,R14,R16, R18,R29,R30,R35	10k
2	R4,R36	47k
2	R6,R31	1M
1	R9	200R
1	R10	910R
1	R11	5k
2	R12,R20	4k7
1	R15	6k8
1	R17	3k3
1	R32	100k
3	R33,R39,R40	470R
1	R34	100R
1	R38	22k
3	C1,C11,C14	10uF
2	C2,C12	1uF
6	C3,C7,C9,C10,C13,C15	100nF
1	C4	220nF
1	C5	470nF
1	C6	220uF
1	C8	100uF
1	U1	1854
1	U2	7109
1	U3	4066
1	U4	LM35
1	U5	4024
1	U6	7805
1	U7	7905
1	U9	4060
1	U10	OP07
1	U11	LM311
3	D1,D3,D5	1N4148
1	D2	LED GR
1	D4	BY164
1	D6	LED RD
1	D7	LED GL
2	Q1,Q2	BC547
1	Y1	2,4576 MHz
1	J1	CON2
1	JP1	HEADER 4X2
1	CON1	CON5
1	J2	CON3
1	K1	AZ8-1CH-12DE
1	S1	C&K 7101
1	F1	500mA
1	F2	50mA
1	T1	E 16 TR 4

Fig 5: stuklijst

De basis subroutines:

```
init_rs   lda   #%00001011
          sta   $E132        acia command
          lda   #$18
          sta   $E133        acia control, 1200 baud
          rts

dat_rs    lda   temp         haal de data op
so        sta   $E130        verzend de data
si        lda   #$08
          and   $E131
          beq   si           wacht op eerste byte
          lda   $E130        lees de high-byte
          sta   input+1      bewaar de high-byte

1         lda   #$08
          and   $E131
          beq   si           wacht op tweede byte
          lda   $E130        lees de low-byte
          sta   input        bewaar de low-byte

          rts
```

Met de reuze kracht van de IBM en IBM-compatibele machines heb ik geen ervaring. Blijkbaar geen schande want de meeste "programmeurs" blijken wat moeite te hebben met het verzenden van een byte-tje langs de COM-poort. QBASIC van DOS-5.0 heeft blijkbaar intern problemen met het omzetten van characters naar getallen bij het ontvangen van de bytes. Dat was toch een (voorlopige ?) conclusie tijdens de bijeenkomst in Krommenie, want verschillende, meer beslagen programmeurs hebben toen mee geprobeerd om in QBASIC een LEDje (bij mij thuis de status van 18 kWatt) aan en uit te kunnen doen en de bytes binnen te halen. Tegen vijf uur was de raad zeer duidelijk: doe het in assembler met INT's of doe het in C.

Mijn eerste gebrabbel in C++ (na eindeloze variaties) heeft reeds het volgende resultaat: zie figuur 6.

In praktijk is het raadzaam om nog enkele controles uit te voeren: zijn de bytes wel goed ontvangen en zijn ze wel alle twee ontvangen?

Frank Vandekerkhove
St-Michielsstr. 4
B - 9130 Verrebroek_Beveren

```
/* RS-232 Thermo Interface                                          */
/*                                                                  */
/* Een teller wordt steeds per seconde opgehoogd.                   */
/* De lsb van de teller wordt via COM1 naar interface               */
/* gestuurd. (d.i. oven aan/uit met PtRh ingang)                    */
/*                                                                  */
/* Instelling: 1200,8N1, no handshaking                             */

#include <bios>.h
#include <conio>.h
#include <stdio>.h

#define COM1          0
#define DATA_READY    0x100
#define TRUE          1
#define FALSE         0

#define SETTINGS ( 0x80 | 0x03 | 0x00 | 0x00)

int main(void)
{
        int high, low, in, DONE = FALSE;
        unsigned char data, a = 0x00;

        bioscom(0, SETTINGS, COM1);          /* set 1200,8N1        */
        cprintf("... use [ESC] to exit ...\n");
        while (!DONE)
        {
        a=a+1;                               /* increase teller     */
        printf("\nTeller = %d ", a);
        delay(1000);
        data = a & 0x01;                     /* get b0              */
        printf("\tdata = 0x%X",data);
        bioscom(1, data, COM1);              /* switch on/off       */
        high = bioscom(2, 0, COM1);          /* get last A/D-conv.  */
        low = bioscom(2, 0, COM1);
        printf("\thigh-byte = %x",high);
        printf("\tlow-byte = %x",low);

        if (kbhit())                         /* key depressed ?     */
                {
                if ((in=getch())=='\x1B')    /*ESC=end of job       */
                DONE = TRUE;
                }
        }
return 0;
}
```

Fig. 6: Listing in C

Intel is de tel kwijt...

Het nieuwste familielid in de Intel processorfamilie heet "Pentium". Dat stinkt naar penta, iets dat met vijf te maken heeft. Toch is dit niet de vijfde Intel processor, want als je de 8008, de 8080, en de 8085 meetelt, is dit de achtste processor uit de Intel-stal, en niet de vijfde. Toch horen die voorgangers er ook bij, want daar liggen de wortels van de processor waar Intel wel mee begint te tellen, namelijk de 8088. Maar ook dan klopt het getalletje penta niet, want dan zou het de zesde processor zijn!

U snapt het natuurlijk al: dat "Pentium" verwijst naar het middelste cijfer in het typenummer van de processor, want de Pentium had eigenlijk de i80586DX moeten worden. Zoals reeds eerder in de μP Kenner te lezen viel heeft Intel geprobeerd om de typenummers van haar processoren tot gedeponeerd handelsmerk te maken, maar daar heeft een Amerikaanse rechter een stokje voor gestoken door te bepalen dat alleen woorden een gedeponeerd handelsmerk kunnen voorstellen, maar getallen niet. Dus kreeg de nieuwe processor een naam en geen nummer. Pentium dus. We gaan eens kijken wat dit allemaal moet voorstellen.

Techniek

De Pentium is ondergebracht op een chip van reusachtige afmetingen: 20 x 20 mm, oftewel 4 vierkante centimeter! Dat komt niet omdat Intel van die grofstoffelijke chips maakt, want de Pentium is in een zogenaamde submicron technologie gemaakt: 0.8 μm BICMOS. De enorme afmetingen worden veroorzaakt door het grote aantal transistoren dat op de chip zit: 3.1 miljoen stuks. De ook niet bescheiden 80486DX had er "maar" 1.3 miljoen...

Ook de Pentium is een heethoofd.

De Pentium praat met de buitenwereld via 273 aansluitpinnen van zijn Pin Grid Array behuizing. Die is weer van het keramische type, want ook de Pentium is een heethoofd. De eerste geruchten over een vroegtijdige hittedood van de CPU zijn er al weer, ofschoon Intel volhoudt dat de processor zonder koeling en koellichaam te bedrijven is, mits de buitenkant van de behuizing onder de 85 graden Celsius blijft.

Die 273 aansluitpinnen zijn onder andere nodig voor de eerste uitbreiding ten opzichte van de 80486: een 64 bits brede databus. De Pentium is dus een 64 bits processor. De adresbus is slechts 32 bit breed, waarbij een wat merkwaardige manier van aansturen is gekozen: A3 tot en met A31 zijn direct naar buiten gevoerd, maar A0 tot en met A2 zitten verstopt in een achttal byte-select signalen die aangeven welke van de acht bytes het geheugen moet aanleveren over de databus.

Prestaties

De Pentium wordt aangeboden met een maximum clockfrequentie van 66 MHz, hetzelfde tempo waarin de snelste 486's (de 80486DX2-66) de racebaan worden rondgeschopt. Toch is de Pentium dubbel zo snel als de snelste 486 als het om integers gaat, en tot zeven maal zo snel bij het consumeren van fluitende punt getallen. Mocht u dit nog niet snel genoeg vinden: de eerste geruchten over een dubbelclockende Pentium, alsmede 100 MHz versies zijn er nu al.

De vraag is nu hoe deze prestaties in hardware worden gehaald.

Parallelprocessing

Dit is sleutel nummer 1 tot de snelheidswinst. Want op de chip zit niet 1, maar zijn twee integer ALU's ondergebracht. De twee ALU's staan volledig parallel, waardoor de processor twee integerbewerkingen per clockcyclus kan uitvoeren. Let wel: 32-bit integerbewerkingen, want de ALU's zijn 32 bit breed.

Om die twee ALU's tijdig van de benodige data te voorzien en ze te vertellen wat er met die data moet gebeuren (de instructie(s)) zitten daarom heen twee 8 kbyte on-chip caches: een code cache en een data cache. Die twee caches vormen sleutel nummer twee tot de prestatiewinst. De data cache staat twee toegangen per cyclus toe, een voor iedere ALU. Intel heeft de caches ook nog een naam meegegeven: de MESI-cache. MESI staat hierbij voor de eerste letters van de vier toestanden waarin een cache-line zich kan bevinden:

Modified: De cache-line is veranderd, maar de bijbehorende geheugenlocaties nog niet. Deze toestand wordt ook wel met "dirty" aangeduid.

Exclusive: De cache-line komt maar in 1 van de twee ALU caches voor, en is nog niet veranderd.

Shared: De cache line bevindt zich ook in de andere ALU cache. Indien de cache-line wordt gewijzigd, wordt zowel de andere kopie, als het hoofdgeheugen door middel van een write-through actie veranderd.

Invalid: De gevraagde data staat niet in de cache of is ongeldig.

De vier letters geven dus de vier toestanden aan waarin de cache-lines zich kunnen bevinden. In de Pentium zit dan ook een state-machine die de cachetoestanden beheert. Iedere stateverandering leidt in principe tot een actie, zoals het ophalen en/of terugschrijven van/naar het hoofdgeheugen of het laden van de cache met data uit de cache voor de andere ALU.

De derde sleutel tot de verbazingwekkende prestatiewinst is de opzet van de mathematische coprocessor of floating point unit. De FPU is in de Pentium voorzien van aparte hardware voor optellen, vermenigvuldigen en delen. Verder is de FPU optimaal gepipelined. Netto resultaat: de FPU is tot 7 maal sneller dan de FPU uit de snelste 486.

En nog zijn de trucs van Intel niet op: de vierde sleutel tot de prestatieverhoging is een zogenaamd branch prediction mechanisme. Dit is op te vatten als een kleine instructiecache, die ervoor zorgt dat de pipeline niet onderbroken wordt door onnodige code prefetches: bij een branch terug is de kans zeer groot dat de opcode waar naar toe gesprongen worden nog in de code cache staat. Dit mechanisme werkt ook bij voorwaardse sprongen: als er gesprongen wordt naar code die nog niet in de cache staat, wordt de codeprefetcher actief en deze zorgt ervoor dat de betreffende code in de cache beschikbaar is voordat de processor deze code daadwerkelijk gaat uitvoeren. Het netto resultaat is dat alle voorwaardelijke sprongopdrachten en alle NEAR-jumps in 1 clockcyclus worden uitgevoerd.

Instructieset

De instructieset is geheel gelijk aan die van de 80486, en is dus ook binair compatibel met de overige 80X86 CPU's. Er zijn slechts drie nieuwe instructies: CMPXCHG8B (ik dacht dat mnemonics altijd maar 3 letters hadden?), CPUID en MOV CR4,r32 respectieve MOV r32,CR4.

De laatste instructie is het snelst uitgelegd: de Pentium heeft een statusregister CR4 extra ten opzichte van de 80486, en heeft derhalve ook een tweetal instructies nodig om dit register te kunnen bereiken. Via dit register kunnen namelijk de extra architectuuruitbreidingen van de Pentium bestuurd worden.

CPUID vertelt via het EAX register welke processor er in het systeem zit. Dit omzeilt een al oud probleem: hoe komt een applicatie er achter wat de processor wel, en niet kan. Voor het eerst kun je dus gewoon vragen wie er de baas is in het systeem.

CMPXCHG8B betekent "compare and exchange 8 bytes". Het vergelijkt de 8 bytes in EDX:EAX met een 8-byte geheugenlocatie. Zijn de 8 bytes aan elkaar gelijk, dan wordt de Z-vlag gezet, en wordt de locatie gevuld met de inhoud van ECX:EBX. Is er een verschil, dat wordt de Z-vlag gereset, en wordt EDX:EAX in de geheugenlocatie geschreven. Dit is speciaal bedoeld voor synchronisatiedoeleinden in multiprocessorsystemen.

Hardware

Nieuw is een INIT-pin. De INIT-pin brengt de processor terug naar real-mode, en lijkt daarom een beetje op een reset. Het is geen reset, want na bediening van de INIT-pin staat de CPU weliswaar in real mode, maar alle register- en ook cacheinhouden zijn onaangeroerd gebleven.

Wat ook nieuw is, is pariteitscontrole op de adressen databussen. De processor doet hier verder niets mee maar genereert wel pariteitfout-signalen, die echter verder door het omliggende syteem moeten worden afgehandeld.

Nieuws?

Niet alles is nieuw en of net uitgevonden: zo had AMD's AM29000 RISC processor al het genoemde branch prediction systeem, dat daar weliswaar een andere naam had (branch target cache) maar met dezelfde functie. Ook de pipelined FPU is niet nieuw: die zat ook al in de i860. Wel nieuw is de combinatie van deze zaken met een 80X86 architectuur op een superchip. Wat wellicht ook te verwachten was is toch niet gebeurd: de Pentium heeft, net als de 8088 nog steeds maar vier general purpose registers. En dat is bij zoveel prestatiegeweld misschien wel een beetje weinig.

In dit verhaaltje is nog lang alles niet over de Pentium verteld. Er is nog veel meer. Maar dat vergt ook veel meer detail, en diepere kennis van moderne computerarchitecturen en de overige Intel processors. Maar u heeft in ieder geval een idee wat er onder de naam Pentium schuil gaat: meer dan twee 486's op 1 chip met nog wat extra's erbij. Het wachten is nu op de Taiwanezen. Op het eerste spotgoedkope moederbord. Als Intel tenminste niet weer smerige trucs uit gaat halen met bewuste schaarsheid en opgeschroefde CPU-prijzen. Want AMD is nog steeds niet klaar met zijn 486-kloon...

Nico de Vries

Literatuur

1. MC april 1993, bladzijde 44 e.v.: Pentium-Prozessor

Een écht document-opmaak systeem: TEX en LATEX

Inleiding

Dit verhaal gaat over TEX, een zogenaamd documentpreparatie systeem. In het bijzonder gaat het over LATEX, een speciale versie van TEX die wat simpeler te gebruiken is.

Hoezo, een documentpreparatiesysteem? Met WP kan ik toch ook een leuk opgemaakt tekstje in elkaar draaien? En met Ventura kan ik toch voor een mooie opmaak zorgen ? En TEX, dat heb ik wel eens gezien, ziet er heel vies uit met al die backslash-commando's in de tekst, en het is helemaal niet WYSIWIG !

Deze stellingname is al ettelijke keren de aanleiding geweest van eindeloze 'flame wars' op de internationale computer netwerken. Ook in Nederland kunnen is de USENET newsgroup `nlnet.misc` berucht om de steeds weer oplaaiende discussies over wat nou beter is, TEX of WP. Het begint altijd met iemand die iets moeilijks met WP wil doen.....

Maar wat is nou TEX precies? En kan ik het eens een keer uitproberen? Daarom dit verhaal in de KIM-Kenner. Tevens ben ik van plan een voordracht te houden op de bijeenkomst in Utrecht op 14 mei 1993. Daar kunnen dan vragen beantwoord worden, en zullen er public domain softwarepakketten beschikbaar zijn.

1 Wat is TEX

TEX[1] is een tekstopmaaksysteem dat is ontwikkeld door Donald Knuth, een beroemd hoogleraar in de informatica aan de universiteit van Stanford in Californië. Hij was zo ontevreden met de toenmalige tekstopmaaksystemen (die er vooral op het gebied van het zetten van formules een potje van maakten) dat hij zijn eigen systeem ontwikkelde voor zijn boekenreeks *The Art of Computer Programming.* Deze reeks behoort tot de standaard-literatuur van iedere informatica student. Tussen 1978 en 1984 groeide TEX in zijn huidige vorm [Knu86a].

TEX is een zet-systeem, dat bijzonder geschikt is voor het vervaardigen van 'druk-rijpe' technische documentatie; vooral dié documentatie die mathematische formules bevat. Het kan natuurlijk ook voor andere teksten gebruikt worden, variërend van brieven tot complete boeken. TEX is noch een Desk Top Publishing-pakket noch een tekstverwerker !! TEX is eigenlijk een programmeertaal, bedoeld voor het zetten van tekst. Hierbij wordt in belangrijke mate gebruik gemaakt van macro-expansie (TEX is eigenlijk een grote macro). Omdat het zich als een programmeertaal presenteert met vele honderden zet-opdrachten, schrikt het vele mensen af om het te gebruiken: je moet ervoor programmeren, alleen geschikt voor wiskundigen en computerfreaks. Die programmeerbaarheid is tevens de grote kracht van TEX: je kunt letterlijk alles op een consistente manier doen met je tekst.

[1] Uit te spreken als 'tech', komt van het Griekse τεχ. Als de gezakte 'E' niet mogelijk is is de officiële schrijfwijze TeX

Macropakketten: LATEX

Omdat TEX zo enorm veel mogelijkheden die eigenlijk best primitief zijn heeft schrikt het beginnende gebruikers af: er is nogal veel 'programmeerwerk' nodig. Dit wordt ondervangen door het aanbieden van macro-pakketten zoals LATEX en AMS-TEX. LATEX[2] is door Leslie Lamport [Lam86] is geschreven en maakt van TEX gebruik. Het stelt de auteur in staat zijn publicaties op eenvoudige wijze en met gebruik van een van te voren opgegeven lay-out, met boekdruk-kwaliteit te zetten en af te drukken.

Auteur, ontwerper en zetter

Bij elke publicatie geeft de auteur de uitgever een (over het algemeen) getypt manuscript. De ontwerper van de uitgeverij beslist dan over de lay-out van de publicatie (regellengte, lettertype, spatiering etc.) Het document draagt dan het stempel van de ontwerper en niet van de auteur. LATEX is te beschouwen als de ontwerper, TEX als de zetter. De LATEX-commando's worden in TEX-commando's vertaald. Een menselijke ontwerper herkent de doelstellingen van de auteur meestal vanuit zijn vakbekwaamheid en de inhoud van het manuscript. LATEX is daarentegen 'alleen maar' een programma en behoeft dus informatie van de auteur, waarmee de logische struktuur van de tekst aangegeven wordt. Deze informatie wordt door middel van zogenoemde 'commando's' binnen de tekst aangegeven.

De DTP pakketten zijn anders van opbouw. Hiermee wordt op 'interactieve' wijze de layout van de publicatie vastgelegd. Op het scherm is precies te zien hoe de pagina er uit gaat zien. Zulke pakketten worden daarom ook wel WYSIWYG pakketten genoemd (What You See Is What You Get). Deze pakketten worden door TEX goeroes ook wel WYSIAWYG, waar de A voor 'all' staat. Er zijn ook veel minder vleiende verbasteringen in omloop, zoals WYSIWYMG, WYSIWYCNG etc. Het kenmerk is wel dat de auteur invloed kan uitoefenen op de layout van de tekst, en dat deze invloeden heel moeilijk weer weg te poetsen zijn door een layouter van een uitgeverij.

[2] uitspraak "Lah-tech" of "Lee-tech", kan ook als LaTeX geschreven worden. Het heeft dus niets met Flexa dekkend wit te maken

Dit speelt vooral bij (wetenschappelijke) tijdschriften een grote rol. Typografisch ontwerpen is een vak, dat men moet (kan) leren. Ongeoefende auteurs maken vaak zware typografische fouten. Vele leken denken, ten onrechte, dat typografie alleen maar een kwestie van smaak is; wanneer een document er 'mooi' uitziet, is het ook goed ontworpen. Daar documenten echter gelezen dienen te worden, zijn 'leesbaarheid' en begrijpbaarheid belangrijker dan het uiterlijk.

De letter-grootte en nummering van titels van hoofdstukken en paragrafen moet zo gekozen worden, dat de de struktuur van hoofdstukken en alinea's duidelijk herkenbaar is. De regellengte dient zo gekozen te worden, dat vermoeiende oogbewegingen voor de lezer voorkomen worden en niet zo, dat het papier zo mooi mogelijk 'gevuld' wordt met letters. Dit laatste wordt ook wel het ontwerp van de bladspiegel genoemd. Met DTP-pakketten maken auteurs over algemeen 'esthetisch mooie', maar slecht leesbare documenten.

Met LATEX ziet de auteur over het algemeen niet hoe de tekst die hij/zij aan het invoeren is er uit zal zien, terwijl hij aan het typen is. Het is wel altijd mogelijk om een proef-afdruk te maken, maar dit kost wat meer tijd: je moet eerst je tekst 'compileren'.

LATEX voorkomt typografische fouten, omdat het de auteur dwingt de logische struktuur van een tekst aan te geven, en geen tijd te besteden aan layout-zaken. Dat soort dingen worden door het macro- pakket geregeld. De benodigde kennis van typografisch verantwoord layouten wordt als het ware overgedragen van de amateur-auteur naar de professionele graficus die het macro-pakket heeft ontworpen. LATEX gebruikt automatisch de gedefiniëerde layout.

In de uitgevers-wereld is men de waarde van TEX in gaan zien: vooral bij de wetenschappelijke uitgeverijen wordt de output van TEX gezien als zetwerk van superieure kwaliteit. Sommige uitgevers gaan zo ver dat zij een eigen macro-pakket aanbieden op basis van TEX. Hierin definiëren zij alle layout-kenmerken van hun tijdschrift, en de bedoeling is dan dat je je aan de regels van dat macro-pakket houdt, en je niet meer zelf met de layout bemoeit. Dit macro-pakket kan dan weer gebaseerd zijn op LATEX, we noemen het dan een style-file. Als je je manuscript hiermee opmaakt, kun je het op floppy (of zelfs E-mail) aanleveren: het gaat dan in een klap naar de foto-zetter en de drukpers. Zo wordt het tijdrovende overtypen en aansluitend 'proofreaden' om de onvermijdelijke fouten uit de formules te halen voorkomen. Komt geen layout-afdeling meer aan te pas. Er zijn wetenschappelijke tijdschriften waar de 'publicatie-cyclus' van anderhalf jaar naar drie maanden is teruggebracht: de papers hoeven alleen nog maar naar de reviewers, die beslissen of het geplaatst moet worden.

Voor- en nadelen

Vergeleken met andere tekstverwerkingspakketten, heeft LATEX de volgende voordelen:

- Er zijn diverse professioneel vormgegeven layouts beschikbaar, waarmee de documenten er inderdaad als 'gedrukt' uitzien.
- Het zetten van mathematische formules wordt bijzonder goed ondersteund. Tot nog toe kan niets daaraan tippen. Zelfs de formule-editor van WP niet. Deze formule-editor is eigenlijk 'gejat' van TEX (de commando's zijn bijna precies hetzelfde), alleen is hij veel minder krachtig.
- De gebruiker hoeft maar een paar, gemakkelijk te begrijpen, commando's te leren. Deze commando's betreffen alleen de logische structuur van het document, de gebruiker hoeft zich nauwelijks bezig te houden met de druktechnische details.
- Complexe strukturen zoals voetnoten, literatuuropgaven, inhoudsopgaven, tabellen etc. en zelfs eenvoudige tekeningen, kunnen zonder al te veel moeilijkheden gemaakt worden.
- Met LATEX kan er op verschillende platforms (Unix, DOS) aan dezelfde tekst gewerkt worden. De invoer-teksten met de commando's zijn platte ASCII, zonder speciale karakters voor opmaak. TEX documenten kun je ook zonder vorm van coderen per E-mail versturen (moet je maar eens met WP-documenten proberen).
- TEX 'compileert' het document naar een printer-onafhankelijk `dvi` formaat. Voor het afdrukken zorgt de printerdriver: die neemt alle printer-eigenaardigheden voor zijn rekening. De printer-drivers zorgen op 'hun' printertypes voor precies dezelfde opmaak: je kunt je document op je 9-naalds printer afdrukken, en vervolgens naar een drukker gaan en daar een grote foto-zetter gebruiken. Afgezien van het kwaliteitsverschil is het resultaat hetzelfde.

  Bij de DTP pakketten lijkt elk pakket alle printers aan te moeten kunnen sturen. Dat heeft tot gevolg dat sommige printers wel eens stiefmoederlijk behandeld worden, of dat het pakket zeer groot wordt (WP!). Ook werkt het uitwisselen tussen meerdere typen printers niet altijd goed: fonts kloppen niet, regels verschuiven t.o.v. elkaar en meer ellende.

LATEX heeft natuurlijk ook nadelen:

- Het systeemverbruik (rekentijd en opslagcapaciteit) is hoger dan bij primitievere tekstverwerkers. Alhoewel sommige DTP pakketten ook niet vies zijn van een megabytje.

- De documenten kunnen alleen op dure laserprinters, want daarop komen ze veel beter tot hun recht. Bovendien gaat dat veel sneller. De 'goedkopere' (matrix) printers voldoen nauwelijks en worden over het algemeen alleen maar voor tussentijdse resultaten gebruikt. 24-pin printers voldoen wel (alhoewel de tekst wat vet lijkt) maar zijn erg langzaam in het afdrukken. Over het algemeen zijn er, vooral voor de kleinere computers als PC's en werkstations, zg. 'previewers' beschikbaar.

- Binnen door LATEX ondersteunde standaard layouts kunnen weliswaar enige parameters gevariëerd worden, maar ingrijpende afwijkingen van de gebruikte layout kunnen slechts met veel moeilijkheden en kennis van zaken tot stand gebracht worden (Ontwerp van een nieuwe 'document-stijl').

- LATEX documenten zien er soms uit als 'eenheidsworst'. Dit komt vooral omdat de auteurs de standaard document-stijlen gebruiken, waardoor het allemaal veel op elkaar lijkt. Een actievere rol van uitgevers om meer stijlen beschikbaar te stellen is gewenst. Verder wordt heel vaak hetzelfde font gebruikt.

- Voor sommig werk (advertenties, posters) is de directe feedback van een WYSIWYG DTP pakket veel gemakkelijker in het gebruik. Bekijk dit onderscheid als het onderscheid tussen een 'compiler' en een 'interpreter'.

Fonts

Teksten die met TEX en LATEX gemaakt zijn zijn meestal te herkennen aan het font dat gebruikt is. Dat font is bijna altijd de 'Computer Modern', ook deze tekst is daarmee gezet. Deze letter is door Knuth ontworpen toen hij TEX ontwikkelde. Hij heeft er zelfs een font- opmaak-programma METAFONT voor geschreven, geheel in stijl met de geest van TEX. Helaas is METAFONT eigenlijk alleen gebruikt om er de Computer Modern mee te maken, hoewel het een heel krachtige font-beschrijvingstaal is, krachtiger dan Postscript Type I. Wel zijn er diverse fonts voor andere alfabetten gemaakt, zoals Cyrillisch en Arabisch.

TEX zelf werkt eigenlijk alleen met zogenaamde `.tfm`-files: hierin staan breedte en hoogte van de letters. METAFONT maakt van de beschrijvingen in `.mf`-files die `.tfm`-files aan, en tevens `.pk`-files, waarin de letters zelf staan. De `.tfm`-files zijn onafhankelijk van de vergroting van de letter. De `.pk`-files worden per letter-grootte aangemaakt (dus niet 'scalable') voor het beste resultaat. De previewer en printerdrivers gebruiken deze om de `dvi`-file af te drukken. In de meeste TEX installaties waar METAFONT aanwezig is wordt de mogelijkheid gebruikt om automatisch de benodigde `.pk`-files te genereren. Alleen de meest gebruikte blijven op de disk staan.

Het is mogelijk om Postscript fonts te gebruiken met TEX, en dit wordt ook hoe langer hoe meer gedaan. De Type I fonts moeten dan wel bewerkt worden voor gebruik met TEX: hiervoor is diverse public domain software in omloop. Ook zijn er goede Postscript emulatoren (Ghostscript) en public domain fonts te verkrijgen.

Public domain

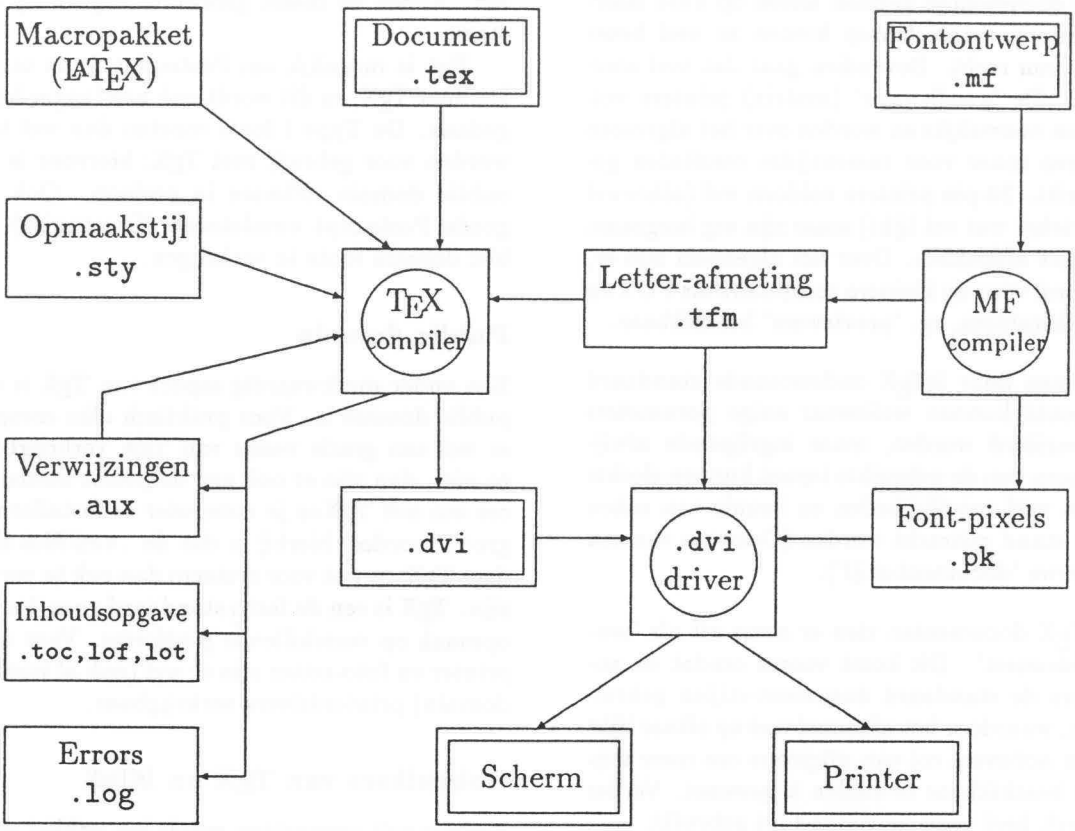
Een ander merkwaardig aspect van TEX is dat het public domain is. Voor praktisch elke computer is er wel een gratis versie van TEX verkrijgbaar, en zo niet, dan zijn er ook nog de public domain sources om zelf TEXop je computer te installeren. Het grote voordeel hierbij is dat de `.tex`-files door iedere TEX op wat voor systeem dan ook te verwerken zijn. TEX is een de facto standaard voor documentopmaak op verschillende platforms. Voor elk type printer en foto-zetter zijn er wel (ook al weer public domain) printerdrivers verkrijgbaar.

Gebruikers van TEX en LATEX

Zoals je zult verwachten wordt het pakket vooral in het wetenschappelijke wereldje gebruikt. Wiskundigen, natuurkundigen en astronomen roemen het pakket om de goede support voor formules. Informatici kunnen zich er tevens heerlijk in uitleven. Maar ook buiten de β wereld begint TEX in zwang te raken: steeds meer promovendi ontdekken TEX als de manier om het proefschrift een gelikt uiterlijk te geven. Verder wordt het gebruikt voor het afdrukken van manuals, hiervoor is zelfs een speciaal macro-pakket `texinfo` dat die manuals ook geschikt maakt voor online raadpleging. Voor het maken van tijdschriften waarbij de auteurs de kopij elektronisch aanleveren is TEX van onschatbare waarde om de opmaak gelijk te houden. Verder is het waardevol in geval van veel wisselende documenten met verwijzingen naar tabel- en hoofdstuknummers: TEX zorgt ervoor dat de nummering altijd goed loopt.

2 Werken met LATEX

De input voor LATEX is een tekst-file. Deze file wordt met een 'gewone' editor aangemaakt en bevat naast de af te drukken tekst ook de commando's die LATEX vertellen hoe de tekst gezet dient te worden. 'Onzichtbare' tekens zoals de spatie en *carriage return* worden door LATEX als een spatie gezien. *Meerdere* spaties of tabs worden als één spatie behandeld. Een lege regel in de tekstfile tussen twee stukken tekst geeft het einde van een alinea aan. *Meerdere* lege regels worden als één lege regel behandeld. Men kan de woord- en regelafstanden dus slechts veranderen met de juiste LATEX-



Figuur 1: Het opmaak-traject in TEX

commando's en **niet** door het toevoegen van meer spaties en/of lege regels. De volgende symbolen hebben een speciale betekenis voor LaTeX, zij kunnen niet zomaar gebruikt worden.

```
$ & % # _ { } ~ ^ " \ | < >
```

De volgende zeven tekens kunnen door er een \ (Backslash) voor te zetten gewoon afgedrukt worden. De overige symbolen en andere bijzonder tekens kunnen, zoals we nog zullen zien, met speciale commando's als accenten, of in mathematische formules afgedrukt worden. De meeste LaTeX-commando's zien er als volgt uit: Of ze beginnen met een backslash (\) en hebben dan een uit louter letters bestaande naam die door één of meer spaties of een bijzonder teken of een cijfer beëindigd worden, óf ze bestaan uit precies één speciaal teken of cijfer. Hoofd- en kleine letters hebben óók in commando's een verschillende betekenis. Als men na een commando een spatie wil hebben moet men {} gebruiken om het commando te beëindigen of een eigen commando gebruiken om de spatie te verkrijgen. Vele commando's hebben parameters die tussen accolades aangegeven dienen te worden. Andere commando's hebben parameters die weggelaten kunnen worden of tussen blokhaken aangegeven moeten worden. Veel commando's hebben varianten die door het tussenvoegen van een ster *aan-* of *uit*gezet kunnen worden. Accolades kunnen ook gebruikt worden om 'groepen' aan te maken. Een commando is 'aktief' zolang de groep of *environment* nog aktief is. Als een groep of environment beëindigd wordt, eindigt ook de werking van het commando. Alles wat achter een procentteken (%) staat (tot aan het einde van de regel) wordt niet gezien door LaTeX. Dit kan door de auteur gebruikt worden om notities te maken die niet gedrukt dienen te worden.

Opbouw

Het eerste commando in een LaTeX-inputfile moet het commando

```
\documentstyle
```

zijn. Daarna volgen definities die voor het hele document gelden. Met het commando `\begin{document}` begint het zetten van het document. Dan volgt de tekst en alle LaTeX-commando's die het zetten van de tekst beïnvloeden. De input wordt afgesloten met het commando `\end{document}`. Als er na dit commando nog input volgt dan wordt dit door LaTeX genegeerd. Een ietwat complexer voorbeeld is in fig. 2 geschetst.

Er bestaan verschillende 'documentstijlen'.

`article` voor artikelen in wetenschappelijk tijdschriften, voordrachten, praktikumhandleidingen, seminars, korte berichten, programmabeschrijvingen, uitnodigingen, etc. De nederlandse lay-out is érg verschillend van de

```
\documentstyle[twoside,kimken]{article}
\author{Phons Bloemen}
\title{Een \'echt
documentopmaaksysteem: \TeX, \LaTeX}
\begin{document}
\pagestyle{kimken}
\bibliographystyle{alpha}
\maketitle
\begin{abstract}
Voorbeeld van een wetenschappelijk artikel
in de nederlandse taal.
\end{abstract}
\tableofcontents
\section{Begin}
Hier komt dan die fabeltastische tekst\dots
\section{Einde}
\dots\ en hier eindigt het.
\bibliography{texbib}
\end{document}
```

Figuur 2: Opbouw van een artikel

amerikaanse, hiervoor zijn aparte 'stijlen' beschikbaar.

`report` voor langere stukken, die uit meer hoofdstukken bestaan, dissertaties, scripties, etc.

`book` voor boeken.

`kimken` is een documentstijl die ik speciaal voor de KIM- Kenner gemaakt heb, om de opmaak na te bootsen. Je zou de hele KIM-Kenner met deze stijl kunnen zetten.....

Tussen de blokhaken kunnen één of meer, door een komma gescheiden, *opties* opgegeven worden. Hiermee kun je verschillende kleine dingetjes aan de gekozen documnetstijl veranderen.

`11pt` voor een 11 punts 'broodletter', etcetera

`twocolumn` om een tweekoloms 'output' te verkrijgen.

`twoside` om het document dubbelzijdig te zetten (de linker en rechter pagina zijn verschillend!)

`dutch` stelt de nederlandse taal in

`a4` om alles netjes op een A4-tje te krijgen. (de amerikaanse papierformaten zijn anders).

Met het `\pagestyle{`*style*`}` commando kan de paginaopmaak vastgelegd worden. De manier van literatuurverwijzen en de opmaak van de referentielijst worden met `\bibliographystyle{`*style*`}` ingesteld.

Tekst zetten

Normale tekst wordt in blokmode, d.w.z. met (linker en rechter) kantlijnen gezet. LATEX breekt regels en pagina's automatisch af. Het doet erg veel moeite om tot mooi opgemaakte regels te komen. Voor elke alinea wordt de best mogelijke verdeling van woorden en regels gezocht en worden, waar nodig, woorden automatisch afgebroken, dit in tegenstelling tot de DTP paketten ,die op regelbasis afbreken. Hierdoor is het resultaat van het afbreken beter. Ook biedt het geruikte afbreekargolrime meer mogelijkheden om tot goed afbreken van woorden te komen. Eén of meer lege regels kenmerkt het eind van een alinea.

Het hangt van de document style af, hoe de alinea's gezet worden. In artikelen, rapporten en boeken worden alinea's weergegeven door het inspringen van de eerste regel. Als neveneffect worden de afstanden van de 'environment' en van de mathematische vergelijkingen veranderd. Met behulp van de in paragraaf 4 beschreven 'environments' is het mogelijk bepaalde stukken tekst anders te zetten. Het commando \\ of `\newline` breekt de regel af zonder een nieuwe alinea te beginnen. Voor uitzonderingsgevallen kan men het afbreken met bepaalde commando's beïnvloeden. Bijvoorbeeld als het afbreekalgoritme ondanks zijn kwaliteiten 'er niet uitkomt'.

Leestekens

In tegenstelling tot typemachines, waar elke punt en komma evenveel ruimte in beslag neemt als een gewone letter, worden in boeken komma's en punten zo dicht mogelijk tegen de vorige letter aangezet ('kerning'). In boeken is het gebruikelijk om bepaalde letterkombinaties anders te zetten dan de enkele karakters.

ff fi fl AV Te... in plaats van
ff fi fl AV Te...

Met LATEX is het mogelijk om accenten en speciale karakters uit een grote verscheidenheid van talen te gebruiken (zie tabel 1). In voorbeelden wordt meestal de letter o gebruikt, maar het is, in principe, mogelijk om op elke letter een accent te zetten. Als er een accent op een i of een j gezet moet worden, dan moet er een 'puntloze'-i (of -j) gebruikt worden. Een dergelijke i (of j) wordt met het commando `\i` (of `\j`) te voorschijn getoverd.

3 Lettergrootten en -typen

Normaal gesproken kiest LATEX de lettergrootte en het lettertype op grond van de commando's, die de logische structuur van de tekst aangeven, en de definities in de documentstijl. In speciale gevallen kan men ook *zelf* het lettertype of de lettergrootte kiezen, zie tabel 2 en tabel 3.

Tabel 1: Accenten en speciale letters

input	output	input	output
\'o	ò	\'o	ó
\^o	ô	\~o	õ
\=o	ō	\.o	ȯ
\u o	ŏ	\v o	ǒ
\H o	ő	\"o	ö
\c o	o̧	\d o	ọ
\b o	o̲	\t oo	o͡o
\oe	œ	\OE	Œ
\ae	æ	\AE	Æ
\aa	å	\AA	Å
\o	ø	\O	Ø
\l	ł	\L	Ł
\i	ı	\j	ȷ
!‘	¡	?‘	¿

Tabel 2: Lettertypes

\rm	normaal schrift (roman)
\bf	**vet schrift (boldface)**
\it	*cursief schrift (italic)*
\sl	*schuin schrift (slanted)*
\sf	'sans serif' schrift
\sc	'CAPS AND SMALL CAPS' SCHRIFT
\tt	`schrijfmachineschrift`
\boldmath	vet schrift in math. mode

Tabel 3: Lettergroottes

\tiny	vreselijk klein schrift
\scriptsize	heel klein schrift (indices)
\footnotesize	klein schrift (voetnoten)
\small	klein schrift
\normalsize	normaal schrift
\large	groot schrift
\Large	groter schrift
\LARGE	heel groot schrift
\huge	reuzegroot
\Huge	reuzegroot

Internationale teksten

Er zijn een aantal commando's die niet in de originele TEX te vinden zijn. De originele versie ondersteunt alleen maar het zetten van engelse tekst. De 'europese' versie ondersteunt in ieder geval meer talen. Elke taal heeft zijn eigen typografische eigenaardigheden. We zullen hier echter alleen maar de nederlandse eigenaardigheden behandelen. Bij het opstarten van LATEX wordt vastgesteld welke afbreekroutines er gebruikt gaan worden. Er kunnen afbreekpatronen voor verschillende talen tegelijk geladen worden, waarbij er speciale commando's gedefinieerd worden om van taal te wisselen. Ook kun je tekst 'taalafhankelijk' zetten. Het selecteren van een bepaalde taal zorgt er ook voor dat dingen als 'Hoofdstuk' vervangen worden door 'Chapter' enzovoort. Dit systeem wordt heel toepasselijk babel genoemd.

4 Globale opmaakinstrukties

Hoofstukken en paragrafen

Het begin van een hoofdstuk of paragraaf wordt met het commando `\section{...}` aangegeven. Er moet dan wel een logische structuur aangehouden worden.

Bij artikelen gaat dat met `\section`, `\subsection` en `\paragraph`. In boeken kan `\chapter` toegevoegd worden. Artikelen kunnen dan eenvoudig als hoofdstuk in een boek opgenomen worden.

De afstand tussen het kopje en de tekst, het nummeren en het lettertype worden door LATEX via de documentstijl zelf vastgesteld.

Het commando `\tableofcontents` genereert de inhoudsopgave. LATEX gebruikt de kopjes en nummering van de *voorlaatste* 'run'. Als er een nieuw hoofdstuk wordt tussengevoegd , moet men LATEX twee keer aanroepen om een juiste inhoudsopgave te verkrijgen.

Met het commando `\label` en `\ref` is het mogelijk om de door LATEX gegenereerde hoofdstuknummers, sectie- nummers, figuur- en tabelnummers en, voor de wiskundigen, nummers van vergelijkingen in de tekst te gebruiken. LATEX vervangt `\ref{...}` door het in de tekst met `\label{...}` gedefiniëerde nummer. Zo kun je een lekker ingewikkeld bewijs opstellen, met verwijzingen over en weer. Dit kan allemaal door elkaar, LATEX zoekt zelf wel uit wat waarbij hoort. Als er geschoven wordt met tekst, wordt er automatisch opnieuw genummerd (dit kost wel een extra 'run').

`\footnote{Dit is een voetnoot}` zet de voetnoten[3] onder aan de pagina en nummert ze automatisch.

[3] dut is een voetnoot

Environments

Speciale stukken tekst, die anders gezet dienen te worden dan in normale blokmode, worden gekenmerkt door zg. 'environments'. Environments hebben de volgende vorm.

`\begin{`*name*`}` *tekst* `\end{`*name*`}`

Environments zijn 'groepen'. Zij kunnen ook 'genest' worden, daarbij moet men wel de goede volgorde aanhouden.

`quote`-environment wordt gebruikt voor kort citaten, benadrukte zinnen en voorbeelden. de tekst springt links en rechts in.

`quotation`-environment wordt gebruikt voor citaten die uit meerdere alinea's bestaan.

`verse`-environment wordt gebruikt voor gedichten en voorbeelden waarbij de regelopbouw belangrijk is. De regels worden door \\ gescheiden, strofen door lege regels.

Lijst-environments Hiervoor geldt dat je met het `\item` commando de verschillende punten kan aangeven. Deze drie typen environments zijn door elkaar te nesten. Daarbij hangt het 'aandachtsteken' of de 'nummersoort' (getallen, letters, romeinse cijfers) af van de nestdiepte.

- `itemize` wordt gebruikt voor eenvoudige lijsten.
- `enumerate` wordt gebruikt voor lijsten met genummerde regels.
- `description` wordt gebruikt voor lijsten met een beschrijvend kopje.

Kantlijn-environments Deze environments spelen met de opmaak van de tekst:

- `flushleft` vult een zin links uit
- `flushright` vult een zin rechts uit
- `center` is voor het zetten van de tekst in het midden van de regel.

De losse zinnen worden door \\ gescheiden. Als men geen gebruik maakt van \\ dan breekt LATEX de zinnen automatisch af.

`verbatim`-environment De zinnen in dit environment worden precies zo gedrukt zoals zij zijn ingevoerd, maar dan in het 'typewriter' lettertype. Dat betekent, dat alle spaties, einde van zinnen en alle speciale tekens zonder enige bemoeienis van LATEX afgedrukt worden. B.v. een (kort) computer-programma laat zich goed in deze environment zetten. Binnen een zin kunnen korte stukken tekst ook 'letterlijk' worden afgedrukt. Een en ander gebeurt door de betreffende tekst tussen `\verb|` en `|` in te sluiten. In plaats van | mag ook een ander teken gebruikt worden.

Plaatjes

Voor de tussen `\begin{figure}` en `\end{figure}` staande tekst (of de met `\vspace` aangegeven ruimte voor het inplakken van een plaatje) wordt automatisch zoveel ruimte gereseveerd, zodat het er kompleet in past zonder dat het over twee pagina's verdeeld wordt. Met `\caption{...}` kan men de tekening een naam geven. Men hoeft alleen de tekst in te voeren, het woord 'figuur' en het (doorlopende) nummer wordt door LATEX ingevuld.

Helaas is de ondersteuning van plaatjes in TEX niet zo goed als je zou verwachten. Dit ligt aan de manier waarop TEX tegen een tekst aankijkt: als een verzameling letters. Simpele plaatjes als stroomdiagrammen kun je nog wel met LATEX commando's maken, maar ingewikkelder illustraties vergen een moeizaam conversieproces. Hierin komt verbetering, en de integratie van Postscript plaatjes verloopt al vrijwel probleemloos. Een andere truc is om je plaatje te converteren naar een TEX font (pk-file). Daarbij doe je net alsof je plaatje uit een aantal (vreemd gevormde en vrij grote) letters bestaat. Met `\label` en `\ref` kan men aan een plaatje in de tekst refereren.

Tabellen

Tabellen worden analoog met tekeningen in een `{table}` environment gezet. Voor het opstellen van de tabel zijn er twee methoden: In de `tabbing`-environment kan men tabulatorstops net als op de typemachine instellen en gebruiken. Het commando \= zet een tabulator-positie, `\kill` betekent dat de 'master-regel' niet afgedrukt moet worden. \> springt naar de volgende tabulator-positie en \\ breekt de regel af.

De `tabular`-environment is voor het zetten van tabellen, waarbij LATEX automatisch de benodigde kolombreedtes berekent en waarin ook uitvullen en hulplijnen toepasbaar zijn. In de parameterlijst van het commando `\begin{tabular}{...}` word het formaat van de tabel opgegeven.

7C0	hex	Zo kan tekst toegevoegd worden
3700	octaal	
11111000000	binair	
1984	decimaal	OK

```
\begin{tabular}[t]{|rl|p{1.5cm}|}
\hline
7C0 & hex &
    Zo kan tekst toegevoegd worden\\
3700 & octaal & \\
11111000000 & binair & \\
\hline
1984 & decimaal & OK \\
\hline
\end{tabular}
```

Mathematische formules zetten

Mathematische 'teksten' binnen een alinea worden tussen \(en \) of tussen $ en $ ingesloten. Als mathematische tekst gelden zowel complete mathematische formules als uit één karakter bestaande variabelen, griekse letters, 'superscript' en 'subscript' en andere speciale tekens. Een simpel voorbeeld als de stelling van Pythagoras kan gezet worden met `$c^{2}=a^{2}+b^{2}$`, dat levert dan op: $c^2 = a^2 + b^2$.

Grotere mathematische formules of vergelijkingen kan men beter op een aparte regel zetten, met \[en \] of in de environment `{displaymath}`. Het zetten in mathematische mode onderscheidt zich van tekst mode op de volgende punten:

1. Spaties en zins-einden hebben geen enkele betekenis, alle afstanden worden aan de hand van de logica van de mathematische formule vastgesteld. Met speciale commando's kunnen ze beïnvloed worden.
2. Lege regels zijn niet toegestaan (mathematische formules moeten in één alinea staan).
3. Alle 'losse' letters worden als variabelen beschouwd en als zodanig gezet (cursief met de bijbehorende afstand). Wil men ook nog normale tekst (recht schrift) gebruiken dan dient men deze tekst in een `\mbox{...}` te zetten.

Symbolen in mathematische formules

In deze paragraaf worden de belangrijkste symbolen die in mathematische formules gebruikt worden kort behandeld. Voor het hele **Griekse alfabet** zijn er commando's beschikbaar: $\lambda, \xi, \pi, \Phi, \Omega$ is gezet met `$\lambda,\xi,\pi, \Phi, \Omega $`

Daarnaast zijn er nog **mathematische symbolen**: van $\in$, $\Rightarrow$ tot ∞.

Exponenten en Indices kunnen met het teken ^ of _ boven dan wel onder een variabele geschreven worden. Het **wortelteken** word met `\sqrt` aangegeven, de n-de wortel met `\sqrt[n]`. De grootte van het wortelteken wordt door LATEX zelf vastgesteld. Voor het zetten van mathematische **'Accenten'** zoals pijlen of slingertjes boven variabelen zijn er verschillende commando's. Een **breuk** (fraction) wordt dmv. het commando commando `\frac{...}{...}` gezet. Voor eenvoudige breuken kan men echter ook de operator / gebruiken.

a_1	`$a_{1}$`
$e^{-\alpha t}$	`$e^{-\alpha t}`
a^3_{ij}	`$a^{3}_{ij}`
$\sqrt{x}$	`$\sqrt{x}$`
$\sqrt[3]{x^2+\sqrt{y}}$	`$\sqrt[3]{ x^{2}+\sqrt{y}}$`
$\frac{x^2}{k+1}$	`$\frac{ x^{2} }{ k+1 }$`
$x^{\frac{2}{k+1}}$	`$x^{ \frac{2}{k+1} }$`
$x^{1/2}$	`$x^{ 1/2 }$`

Langere slingers en dakjes, die zich over meerdere (tot 3) karakters uitstrekken, verkrijgt men met behulp van de commando's `\widetilde` en `\widehat`.

De commando's `\overbrace` en `\underbrace` zetten een **horizontale accolade** boven danwel onder de aangegeven tekst.

$$\underbrace{a+b+\cdots+z}_{26}$$

```
$\underbrace{ a+b+\cdots+z }_{26}$
```

Het **integraalteken** verkrijgt men met `\int`, het **sommatieteken** met `\sum`, en het **limietteken** met `\lim`. De boven- en ondergrenzen worden met ^ danwel _ aangegeven.

Normaal gesproken worden de grenzen naast het integraalteken gezet (om ruimte te besparen), met het commando `\limits` wordt bereikt dat de grenzen boven en beneden het integraalteken gezet.

$$\sum_{i=1}^{n} \quad \lim_{x\rightarrow\infty}\frac{\sin x}{x} \quad \int_0^{\frac{\pi}{2}} \quad \int\limits_{-\infty}^{+\infty}$$

```
\begin{displaymath}
\sum_{i=1}^{n}
\lim_{x\rightarrow\infty}\frac{\sin x}{x}
\int_{0}^{\frac{\pi}{2}}
\int \limits_{-\infty}^{+\infty}
\end{displaymath}
```

Gebruikt men het commando `\left` voor haakjes openen en het commando `\right` voor haakjes sluiten, dan wordt automatisch de juiste grootte gekozen. Wil men *geen* linker- of rechterhaakje dan gebruikt men het `\left.` of het `\right.` commando.

$$1+\left(\frac{1}{1-x^2}\right)^3$$

```
\begin{displaymath}
1 + \left( \frac{1}{ 1-x^{2} }
   \right) ^3
\end{displaymath}
```

Wil men puntjes hebben (bv. 1,2,...,n) dan kan men gebruik maken van de commando's `\ldots` en `\cdots`. `\ldots` zet de puntjes op de basislijn (low) `\cdots` zet de puntjes in het midden (centered). Daarnaast zijn er ook nog de commando's `\vdots` voor verticale en `\ddots` voor diagonale puntjes. Voor **matrices** e.a is er de `array`-environment, die net zoals de `tabular`-environment functioneert.

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots \\ x_{21} & x_{22} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

```
\begin{displaymath}
{\bf X} =
\left( \begin{array}{ccc}
x_{11} & x_{12} & \ldots \\
```

```
x_{21} & x_{22} & \ldots \\
\vdots & \vdots & \ddots
\end{array} \right)
\end{displaymath}
```

Voor **meerregelige** formules of stelsels van vergelijkingen gebruikt men de environment `eqnarray` en `eqnarray*` inplaats van `equation`. Bij `eqnarray` krijgt elke regel zijn eigen verglijkingsnummer bij `eqnarray*` wordt, net zoals bij `displaymath`, *geen* nummer toegevoegd. Voor stelsels van vergelijkingen die één gemeenschappelijk nummer dienen te hebben, kan men een `array`-environment binnen de `equation`-environment gebruiken. Met der `\label` en `\ref` commando's kan naar de nummers gerefereerd worden.

$$f(x) = \cos x \tag{1}$$
$$f'(x) = -\sin x \tag{2}$$
$$\int_0^x f(y)dy = \sin x \tag{3}$$

```
\begin{eqnarray}
f(x) & = & \cos x       \\
f'(x) & = & -\sin x     \\
\int_{0}^{x} f(y)dy &
 = & \sin x
\end{eqnarray}
```

Zie voor verdere informatie over Math-mode, en voor lijsten met alle te gebruiken tekentjes het LATEX-*Manual* [Lam86]. Ook bestaat er het $\mathcal{AMS}$-TEXpakket, dat geheel op wiskunde is toegespitst. Het bevat nog veel meer leuke wiskunde-tekentjes en speciale environments om ingewikkelde formules af te drukken. Het afdrukken van hele ingewikkelde formules is ook met LATEXgeen punt.

5 De software

Utility programma's

Bij TEX en LATEX horen een hele serie hulpprogramma's, die ik hier in het kort behandel:

BibTEX is een programma om literatuur-lijsten mee aan te maken. Dit programma werkt met literatuur-databases (`.bib`-files) en stijldefinitiefiles (`.bst`-files) die aan geven hoe de literatuurlijst moet worden opgemaakt. Er is een programma **BiBDB** om je literatuurbestand mee te onderhouden.

MakeIndx dient voor het genereren van indexen.

TEXCAD is een heel simpel tekenprogrammaatje. Het is geschikt om stroomdiagrammen etc in LATEXpicture formaat te maken.

Grafische omzetprogramma's in alle soorten en mate om diverse soorten plaatjes in je TEX document op te nemen. Voor het verwerken van Postscript plaatjes is de public domain interpreter Ghostscript wijd verbreid.

Met BM2FONT zet je elk plaatje om naar een ad-hoc TEX 'font': je doet net alsof het plaatje uit hele grote, vreemde letters bestaat.

METAFONT is een programma om fonts mee te ontwerpen. Dit is op dezelfde leest geschoeid als TEX: het is dus niet zo'n programma om met muis en windows lettertjes te tekenen, maar een 'programmeertaal' voor fonts. Je kunt dan lettertypen ontwerpen die zich aanpassen aan de afdruk-grootte (kleine letters worden iets 'dikker'). Jammer genoeg zijn er maar weinig echte METAFONTs. De Computer Modern letter is de belangrijkste. Er zijn conversieprogramma's die Postscript Type I fonts omzetten naar iets waar TEX mee overweg kan (`.mf`-files, of `.pk`/`.tfm` files).

Previewer waarme je de `.dvi`-files na het 'compileren' maar voor het afdrukken even op het scherm kan bekijken. Een goede previewer is heel nuttig, en spaart een hoop papier.

Printerdrivers om de `.dvi`-files af te drukken. Beschikbaar voor vele printers.

TEX voor de PC : emTEX

Voor de PC is er een fantastische implementatie van TEX beschikbaar: emTEX. Dit pakket is public domain en bevat een TEX compiler met alle utilities. Onder andere een hele mooie previewer `dviscr`, en printerdrivers voor Laserjet en matrixprinters. De TEX en METAFONT compilers zijn zelfs in speciale, protected mode 386 versies beschikbaar. Dit hele pakket is door Eberhard Mattes ontwikkeld. Hij heeft er ook heel informatieve handleidingen voor geschreven.

Als je alles wil hebben, reken dan op een taartpunt ter grootte van 15 Mb van je harde schijf. Hierbij zit dan ook een complete set fonts voor Laserjets (met METAFONT kun je die zelf genereren, maar dat kan aardig wat tijd kosten). 'Gestripte' versies (geheel functioneel, alleen gestript wat betreft sommige fonts en bepaalde utilities) nemen 2.5 Mb in beslag. Er zijn ook speciale 'shells' voor TEX: geïtegreerde TEX opmaak-omgevingen die soms verdacht veel op de Turbo-compilers lijken.

Dit pakket is op het Internet op vele plaatsen te verkrijgen (anonieme FTP doet wonderen, probeer `sol.cs.ruu.nl` of `ftp.stack.urc.tue.nl` maar). Het pakket is ook beschikbaar op het BBS van de club, in de speciale TEX area. Bij ondergetekende kan eventueel een set floppen besteld worden.

Linux

Wat Linux betreft: omdat dit een Unix variant is, kun je zelf je TEX maken met behulp van de officiele

TEX sources. Ook weer met anonFTP op het net 'op te vissen': `ftp.win.tue.nl` is een goede Linux site. Bij uigebreidere Linux distributies wordt TEX meegeleverd. Er wordt naar gestreefd ook Linux versies op het BBS te zetten. Overigens kunnen style-files, fonts, en macro-pakketten op elk willekeurig platform gebruikt worden. De genoemde ftp-servers bevatten naast de specifieke PC en Linux implementaties ook implementaties voor andere platforms, en een schat aan platform-onafhankelijke style-files, macro-pakketten en source code.

TEX gebruikersclubs

Voor TEX is een hele cultuur van clubs onstaan. Een greep:

- NTG (Nederlandstalige TEX Gebruikersclub), Postbus 394, 1740AJ Schagen. Zij geven de MAPS uit: het nederlandse TEX tijdschrift, en organiseren 2x per jaar een bijeenkomst. Lidmaatschap 75 gulden, studenten 50 gulden.
- TUG (TEX Users Group), de internationale (amerikaanse) groep. Zij geven de TUGboat uit, dat op een echt wetenschappelijk tijdschrift lijkt. Er zijn ook elk jaar conferenties. Lidmaatschap ongeveer 50 dollar.
- `TEX-NL` een distributielijst op het Internet (zogenaamde mailing list).
- `comp.text.tex` een USENET newsgroup.

Verder is er natuurlijk de nodige literatuur verschenen.

Referenties

[Doo90] Michael Doob. *A gentle introduction to TEX*. University of Manitoba, Winnipeg, Canada, 1990. Vrij kopiëerbare file `gentle-i.tex`. Introductie tot TEX.

[Eij91] Victor Eijkhout. *TEX by Topic*. Addison-Wesley, Massachusetts,USA, 1991. Voor gevorderden in TEX.

[PS91] Hubert Partl, Elisabeth Schlegl. *LATEX-Kurzbeschreibung*. Universität Wien, Wien, Österreich, 1991. Handleiding bij emTEX `lkurz.tex`. Vertaald naar Nederlands door André van der Vlies en Piet van Oostrum.

[Knu86a] Donald E. Knuth. *The TEXbook*. Addison Wesley, Massachusetts, USA, 1986. Originele beschrijving van TEX.

[Knu86b] Donald E. Knuth. *The METAFONTbook*. Addison Wesley, Massachusetts, USA, 1986. Originele beschrijving van METAFONT.

[Kop91] Helmut Kopka. *A Beginner's book of TEX*. Springer Verlag, New York, 1991. (Engels, Duits), leerboek.

[Lam86] Leslie Lamport. *LATEX, a document preparation system*. Addison Wesley, Massachusetts, USA, 1986. Originele handleiding van LATEX.

[Mal92] Gavin Maltby. *An introduction to TEX and friends*. UNP, 1992. Vrij kopiëerbare file `latex-n.tex`. Introductie to LATEX.

[RdB88] J.R. Luyten R. de Bruin, C.G. van der Laan. *Publiceren met LATEX*, volume 19. CWI Syllabus, Amsterdam, 1988. Een goede start, boekje kost 25 gulden.

[Sch87] N. Schwartz. *Intrduction to TEX*. Addison-Wesley, Massachusetts, USA, 1987. (Engels, Duits).

[Spi86] M. Spivak. *The Joy of TEX*. AmericaN Mathematical Society, Providence, USA, 1986. Beschrijving van $\mathcal{AMS}$-TEX.

Phons Bloemen
tel. 040-815731
phons@ei.ele.tue.nl

Stoeien met PostScript, deel 3

De vorige keer hadden we het over een herhalingsstructuur. Deze keer gaat het over procedures en het doorgeven van parameters.

Drie op een rij

PostScript, Forth & de HP calculatoren maken alle drie (op een rij) gebruik van een 'stack' om parameters door te kunnen geven. Opdrachten dienen in 'Reversed Polish Notation' (RPN) aangeboden te worden. Zo zal bijvoorbeeld, de opdracht **12 3 mul** twaalf met drie vermenigvuldigen en het resultaat achterlaten op de stack.

Variabelen & procedures

/dikte 0.25 def kent de waarde een kwart toe aan de variabele **dikte**. In het PostScript boek (zie literatuurlijst: 1) staat de officiële syntax (**key value DEF**). De definitie van een procedure gebeurt op vrijwel dezelfde manier, echter nu wordt inhoud van de **proc** ingesloten door accolades **{}**. Een voorbeeld hiervan is: **/cm {72 mul 2.54 div} def**. Dit is de omrekening van de standaard unit van 1/72 inch naar centimeters. **/breedte 21 cm def** geeft zo dus de juiste waarde aan de variabele **breedte**, ingevoerd in centimeters, in standaard units.

In het PostScript boek staat de officiële syntax.

Operators

De parameters die aan een procedure worden meegeven en die we van een procedure terugkrijgen, worden via een stack doorgegeven. Om de volgorde van en het aantal parameters te kunnen manipuleren zijn hiervoor de zogenaamde stack operators. In fig. 1 staan de operators die deze keer gebruikt worden. Een kleine uitleg van de syntax wordt aan de hand van **DUP** gegeven: eerst stond er **elksoort** bovenaan op de stack na het uitvoeren van **DUP** staat er **elksoort** en **elksoort**, het duplicaat. Als er een '-' voor de operator staat, dan verwacht die operator geen parameter op de stack. '-' achter de operator betekent dat de operator geen resultaat op stack terug geeft.

Ruitjespapier

De overgang van lijntjespapier naar ruitjespapier hoeft niet zo groot te zijn, behalve lijnen van links naar rechts te zetten, moet je namelijk ook nog lijnen van onderen naar boven zetten. In het programma van vorige keer hoeft dan ook alleen maar na de eerste lus een tweede lus worden toegevoegd. Toen hadden we echter nog geen procedure waar parameters mee uitgewisseld moeten worden. Nu is er een procedure die een half ruitje, bestaande uit een horizontaal lijntje en een verticaal lijntje, tekent. Als er naast en er boven ook zo een half ruitje wordt getekend, is het eerste ruitje compleet.

De stapel

Wat er allemaal op de parameter stack gebeurt zal ik uitleggen aan de hand van een **ruitbreedte** van 2 een **ruithoogte** van 1, **links** zetten we op 4 en **on-**

Name	Description	
any	Value of any type	
num	Number(integer or floating point)	
Operand Stack Manipulation Operators		
any	DUP any any	duplicate top element
any	POP -	discard top element
any1 any2	EXCH any2 any1	exchange top two elements
Path Construction Operators		
-	NEWPATH -	initialize current path to be empty
x y	MOVETO -	set current point to (x,y)
dx dy	RMOVETO -	relative MOVETO
x y	LINETO -	append straight line to (x,y)
dx dy	RLINETO	relative LINETO
-	STROKE -	paints a line following the current path
Arithmetic and Math Operators		
num1	NEG num2	negative of num1

Fig. 1: operators in PostScript

```
%
%ruitjes papier
%
%Constanten en omrekeningen hiervan
/cm {72 mul 2.54 div} def
%uitgegaan van A4 formaat
/breedte 21 cm def
/hoogte 29.7 cm def

%Declaratie & Initalisatie van variabelen
/ruithoogte 1 cm def
/ruitbreedte 1 cm def
/dikte 0.25 def
/links 0 def
/onder 0 def
/rechts breedte def
/boven hoogte def

%lijndikte instellen
dikte setlinewidth

/HalfRuitje
{
%top van de stack is nu de X-pos., gekregen van 'for'
  exch                    % verwissel X- & Y-positie
%op de stack: de plaats van de hoek van de 'L'
  newpath                 % begin een nieuw 'path'
  moveto                  % absolute plaats bijwerken
  0 ruithoogte rmoveto      % naar boven
  0 ruithoogte neg rlineto  % dan weer omlaag
  ruitbreedte 0 rlineto     % dan naar rechts
  stroke                % en 'stempel' de 'L'
  dup                   % maak weer een copie van Y-
positie
} def

/RegelHalveRuitjes
{
dup                     % Y-positie copieren want
                        % krijgen hem maar 1 keer
                        % is echter vaker nodig
links ruitbreedte rechts {HalfRuitje} for
% teken een regel halve ruitjes
  pop             % verwijder de Y-positie copie
                  % aangemaakt door HalfRuitje
  pop             % verwijder de Y-positie copie
                  %aangemaakt door RegelHalveRuitjes
} def

onder ruithoogte boven {RegelHalveRuitjes} for

showpage % om het resultaat op papier te krijgen

%End of File
```

Fig. 2: ruitjespapier in PostScript

der op 3. De eerste regel geeft de stack en de volgende regel waarom de stack er zo uitziet.

```
(Er is nog niets gebeurd)
3       (door voor het eerst RegelHalveRuitjes aan
         te roepen met for)
3 3     (door de dup in RegelHalveRuitjes)
3 3 4   (door voor het eerst HalfRuitje aan te roepen
         met for)
3 4 3   (door exch in HalfRuitje)
3       (door de moveto die 2 parameters van de
         stack afhaalt)
```

De regels met **Rmove** en **Rlineto** spelen zich lokaal binnen **HalfRuitje** af

```
3 3     (door dup aan het einde van HalfRuitje)
3 3 6   (door de aanroep van HalfRuitje met for)
3 6 3   (door exch in HalfRuitje)
3       (door de moveto die 2 parameters van de
         stack afhaalt)
```

De regels met **Rmove** en **Rlineto** spelen zich lokaal binnen **HalfRuitje** af

```
3 3     (door dup aan het einde van HalfRuitje)
```

En dit gaat dan door tot we aan de rechter kant zijn.

```
3       (door de eerste pop in RegelHalveRuitjes)
-       (door de tweede pop in RegelHalveRuitjes)
4       (door aanroep van RegelHalveRuitjes met for)
4 4     (door de dup in RegelHalveRuitjes)
```

Dan kan er weer een hele regel afgehandeld worden. En die regels worden weer herhaald tot bovenaan.

PostScript is interessant, maar ik heb ook andere interessante projecten, zoals KGN68k.

Geert Stappers (04781-41279)

Literatuur overzicht:

1: PostScript Language Reference Manual, second edition; Adobe Systems Incorporated; Addison-Wesley; ISBN 0-201-18127-4

2: Handboek voor PostScript Programmeurs; David A. Holzgang; Kluwer; ISBN 90-201-2360-2

3: Text und Grafik seitenweise; Tilo Rossmanith, C'T November 1991, Heize Verlag

Nogmaals Programmeertalen

Naar aanleiding van mijn artikeltje over programmeertalen in μP Kenner 80, kreeg ik van Hans van Boheemen uit Son en Breugel een reactie. Dacht ik dat ik, door het SQL-voorbeeld regelrecht uit een boek over te kopiëren, er zeker van kon zijn dat ik geen onzin vertelde, schrijft hij me dat het voorbeeld niet voldoet aan de SQL-standaard! Omdat zijn reactie een aantal leerzame zaken bevat, hierbij een deel van de inhoud van zijn brief:

Onder het hoofd: "Vierde generatie: SQL en aanverwante talen" schrijf je dat deze talen werken in combinatie met een database pakket. Dat is juist en is een gevolg van het feit dat een vierde generatie taal de gebruiker in staat stelt te vragen **wat** hij/zij wil hebben. De computer, of beter gezegd, het programma dient zelf uit te zoeken hoe de data benaderd moet worden en welke delen en in welke vorm dit aan de gebruiker teruggegeven moet worden. Dit betekent dat er zeer strakke regels en afspraken moeten worden gevolgd waarmee de vraagsteller de vraag stelt. Eigenlijk zou je kunnen zeggen dat een groot gedeelte van de intelligentie van de gebruiker nu is ondergebracht in het programma.

Het voorbeeld voldoet niet aan de SQL-standaard!

Vervolgens noem je SQL als gestandaardiseerde taal en geef je in figuur 2 een voorbeeld. Helaas is dit voorbeeld **NIET volgens de genormaliseerde SQL regels!** Sommige bouwers van Relationele Data Bases staan dit wel toe maar houden zich daarmee niet aan internationale SQL regels.

De wereldwijd geaksepteerde norm voor SQL is vastgelegd in de ANSI standaard. Alle grote DBS ontwikkelaars zoals Oracle, IBM en vele andere hebben zich hieraan gebonden en de rest volgt schoorvoetend. Een recent voorbeeld is DBASE; dit bedrijf aksepteert nu ook de ANSI standaard naast hun 'eigen' norm.

Om nog even terug te komen op je voorbeeld; kennelijk is hier bedoeld een query te maken over twee tabellen (GROEPEN en SALARISSEN). Een query over meerdere tabellen vormt een deelverzameling. Meestal wil men in deze deelverzameling de elementen groeperen welke beide verzamelingen gemeenschappelijk hebben. In SQL geschreven dient daarom door de vraagsteller te worden aangegeven welke elementen uit beide verzamelingen gelijk moeten zijn. Daarnaast mag de vraagstelling worden uitgebreid met selekties welke gelden voor elke tabel afzonderlijk.

Het is dus nodig dat beide tabellen element(en) bevatten welke data vertegenwoordigt op basis waarvan een vergelijking tussen beide tabellen kan worden gemaakt. De kolommen waarin dit soort data staat wordt vaak aangeduid met de kreet: 'sleutel kolom'. Een sleutel kolom zal, indien op de juiste wijze genormaliseerd, unieke waarden bevatten en nooit lege velden vertonen. Overigens zijn deze regels niet noodzakelijk. Een sleutel in een tabel is, volgens de ANSI afspraken NIET bekend aan het Relationele DataBase systeem maar 'leeft' slechts in de logische gedachtengang van de gebruiker.

Sommige bouwers van Relationele DataBase systemen hebben een voorziening ingebouwd waarmee sleutel kolom(men) in een tabel worden vastgelegd. Automatisch is zo'n Relationele Database systeem (=DBS) nu in staat de relatie tussen meerdere tabellen te leggen. In die situatie beschrijft de gebruiker niet de koppeling tussen de tabellen - het systeem doet dat zelf. Een dergelijk systeem beperkt de gebruiker omdat de relatie nu voor altijd is vastgelegd in het systeem en is daarom NIET toegestaan volgens de ANSI standaard.

Laat ik een voorbeeld volgens de ANSI regels geven:

Tabel: GROEPEN:

nr	naam	groep
1234	jansen	10
1235	pieters	10
1236	hanse	12
1237	spoek	13

Tabel: SALARISSEN

nr	bedrag	schaal	commissie
1235	1234.00	a12	-
1234	2345.00	a56	234.00
1237	1345.23	c21	-
1239	2121.00	b11	-
1240	2222.00	b11	333.00

Laten we aannemen dat je voorbeeld slaat op de volgende twee tabellen: (de 'sleutel kolom' heet: NR in beide tabellen)

De query zou er nu als volgt uit moeten zien:

```
SELECT      NR, NAAM, BEDRAG
FROM        GROEPEN, SALARISSEN
WHERE       GROEPEN.NR = SALARISSEN.NR
            AND BEDRAG BETWEEN 1000 AND 2000
ORDER BY    NR, NAAM
```

Het resultaat ziet er dan als volgt uit:

nr	naam	bedrag
1235	pieters	1234.00
1237	spoek	1345.23

Duidelijk is te zien dat er nu een vergelijking wordt gemaakt tussen beide tabellen (WHERE GROEPEN.NR = SALARISSEN.NR) waarmee alle gemeenschappelijke elementen uit beide tabellen worden geselekteerd. Daarmee is de deelverzameling een feit. Bovendien wordt een extra selektie gemaakt op de kolom 'bedrag' uit tabel SALARISSEN: (AND BEDRAG BETWEEN 1000 AND 2000).

Jouw voorbeeld bevat nog iets twijfelachtigs: de vraagstelling 'WHERE BEDRAG BETWEEN LAAG AND HOOG' mag wel maar elk DBS dat werkt volgens de ANSI norm zal nu 'LAAG' en 'HOOG' opvatten als een naam van een kolom (uit één van beide tabellen). Aangezien de kolom BEDRAG numerieke data bevat moet de vergelijking ook worden gehouden tegen numerieke waarden. Daarom heb ik 'hoog' en 'laag' vervangen door twee getallen; zouden 'hoog' en 'laag' inderdaad bestaande kolommen zijn uit de tabellen 'SALARISSEN' en 'GROEPEN' dan zouden deze kolommen ook numerieke waarden moeten bevatten.

Wil men alfa-numerieke waarden onderling vergelijken dan zal dit in zijn algemeenheid worden opgesloten tussen enkele of dubbele aanhalingstekens bijvoorbeeld:

```
SELECT      NR, NAAM, BEDRAG
FROM        GROEPEN, SALARISSEN
WHERE       GROEPEN.NR = SALARISSEN.NR
            AND NAAM = "pieters"
ORDER BY    NR, NAAM
```

Tot slot nog enkele kanttekeningen. Een relationele DataBase wordt ook veel gebruikt voor opslag van grafische data (bijv. CADAM) of voor opslag van data, bedoeld voor besturing van robots of numerieke machines! En dan nog dit: SQL wordt **ALTIJD** gebruikt in combinatie met een Relationeel DataBase systeem. In dit opzicht wijkt SQL niet af van bijv. 'C' of Basic of een andere taal welke immers ook alleen kan werken indien aangeboden aan een passende compiler, interpreter of assembler.

w.g. Hans van Boheemen

Naschrift:

Uiteraard ben ik Hans zeer dankbaar voor zijn aanvullingen. Zoals al opgemerkt, had ik het voorbeeld overgenomen uit een SQL leerboek, zonder daarbij de hele beschrijving van de onderliggende tabellen mee over te nemen. Mijn doel was alleen een stukje SQL af te drukken om te laten zien hoe iets dergelijks er uit ziet. Het was echter wel de bedoeling correct SQL te presenteren en dat is dus niet helemaal gelukt.

Wat uit de brief van Hans ook naar voren komt, is dat er veel meer over databases en SQL te vertellen valt dan we in enkele pagina's μP Kenner kwijt kunnen. Neem alleen maar het feit hoe je een database kunt ontwerpen. Daarvoor bestaan technieken met hun eigen theorie en notatie-vormen. Sommige lezers hebben misschien wel eens gehoord van de term "Data Modelling" of "Entity Relation Diagrams". Indien er belangstelling voor bestaat kan daar in de μP Kenner wel eens wat dieper op in gegaan worden.

Wat ook tussen de regels naar voren komt, is het feit dat de theorie van een al dan niet relationele database zwaar leunt op de moderne wiskunde. Hans praat over elementen en (deel-)verzamelingen; zaken die we vroeger op de middelbare school (pré-mamoetwet) niet leerden. Ook daarover valt nog wel eens iets te schrijven. Blijft alleen de vraag of de lezers ook in dergelijke onderwerpen geïnteresseerd zijn. Mocht dit zo zijn, dan valt er waarschijnlijk best iets leuks met dit onderwerp te doen, met voorbeelden in bijvoorbeeld DBASE. Zoals altijd geldt ook weer in dit geval: Laat het mij eens weten; tenslotte bepalen de leden wat er binnen de KGN gebeurt.

Gert van Opbroek.

Eerste les "C"

Inleiding

In de μP Kenner van februari 1993 werd een oproep gedaan voor het verzorgen van een cursus "C". Ik heb daarop gereageerd met de gedachte dat ik al de jaren dat ik lid van de club ben, nooit een bijdrage heb geleverd. (Misschien breng ik nu andere slapende mede-clubleden op een idee?)

Ik ben docent informatica bij een grote computerfirma en geef regelmatig cursussen in de "C" taal. Normaal verzorg ik deze cursus in klasverband maar dat gaat wat moeilijk en daarom heb ik besloten de cursus zodanig te herschrijven en aan te passen dat er een schriftelijke cursus ontstaat. Dit is de eerste pennevrucht.

Zo begon het...

De taal "C" is eind 60-er jaren ontwikkeld door Denis Ritchie die werkte in het Bell Laboratorium. Hij trachtte een taal te ontwikkelen waarmee het gemakkelijk mogelijk zou zijn een universeel Operating system te ontwikkelen voor de toenmalige computersystemen. Intussen weten wij dat hij niet in zijn opzet is geslaagd; toch heeft hij het nageslacht iets nagelaten - de taal "C".

Het pre-historische "C" heette geen "C", maar "A". Later kwam er een nieuwe versie uit die door Ritchie "B" werd genoemd en weer later werd dit "C". Vanaf dat moment is de naam van deze compiler altijd "C" gebleven en werd het min of meer officiële systeem van release nummers gevolgd.

"C" wordt door vrijwel alle grote S/W ontwikkelaars gebruikt voor het ontwikkelen van Operating Systems en programma's.

Ritchie was een Assembler programmeur hetgeen niet ongebruikelijk was in die jaren. Hij kwam op het idee een soort van "editor" te ontwikkelen waarmee op een gemakkelijke manier verschillende assembler source modules aan elkaar gezet konden worden. Denis sloeg de assembler source modules op in een bibliotheek, gaf elk module een unieke naam en kon, door middel van zijn "editor" naar verkiezing een aantal modules met elkaar verbinden en opslaan zodat een nieuw (assembler) source programma ontstond. Daarbij was hij ook in staat extra, nieuwe assembler coding te schrijven en toe te voegen. Daarmee kon hij veel sneller dan in die tijd gebruikelijk was, nieuwe programma's schrijven. Ik ben in het bezit van een heel oude "C" versie welke nog volgens dit principe werkt.

Hoe zit deze cursus in elkaar?

In zes afleveringen leert u:

- De basisbegrippen van "C"
- Syntax regels
- Het schrijven van karakters naar het scherm en het lezen van informatie van een toetsenbord
- Idem maar nu met betrekking tot strings. Rekenen en vergelijkingen
- Het compileren van programma's en het tracen van programma's (zoeken naar fouten in programma's)
- Andere vergelijkingen ontwikkelen en schrijven
- De manier waarop strings zijn opgeslagen en het werken daarmee
- Het werken met files
- Functies en het zelf schrijven daarvan

Elke aflevering bestaat uit theorie en praktijk. De praktijk zult u zelf moeten doen op uw eigen systeem. Daarbij zal blijken dat oefening kunst baart...

"C" is een taal waarmee u op verschillende niveau's kunt werken; dat wil zeggen een programma schrijven op eenvoudige wijze, haast op het niveau van een derde generatie taal. U kunt echter ook programma's ontwikkelen bijna op Assembler niveau, dus heel dicht bij de machinetaal. In deze cursus houden wij ons uitsluitend bezig op "hoog" niveau en zal ik uitgaan van startniveau "nul". Zet daarom de blik op oneindig en de gedachten op nul en begin met deze eerste les.

Basisstructuur

In begin lijkt "C" erg moeilijk (is het ook!). Dat komt vooral door de wijze waarop statements worden gebruikt. Dat lijkt op algebraïsche vraagstukken. U leert al snel hier doorheen te kijken. Daarnaast is deze taal in het begin nogal onoverzichtelijk vanwege het grote aantal "functies". Dit aantal loopt in de honderden en is bovendien afhankelijk van de leverancier van het "C" pakket dat u hebt - of wilt aanschaffen.

Alle "C" programma"s zijn verdeeld in **"FUNCTIONS"**. Een functie in "C" is vergelijkbaar met een subroutine in BASIC of een functie in Pascal. Later komen wij hierop nog terug; voorlopig onthouden wij dat elk "C" programma begint met de **MAIN()** functie, meestal gevolgd door een of meer andere sub-functies. Er is altijd 1 hoofd-functie en soms 1 of meer sub-functies. Een functie bevat een of meer statements, die binnen de functie moeten worden uitgevoerd. Deze statements worden opgesloten tussen: **"DELIMITERS"** Anders gezegd; een blok code dat bij elkaar hoort binnen 1 functie, wordt opgesloten tussen een begin en een eind delimiter.

Een voorbeeld:

```
#include <stdio.h>
#include <stdlib.h>

void main(void)
{
   printf("Ik vroeg om thee, verdorie!");
}
```

Verklaring van de onderdelen

(de nummers komen overeen met de regels)

1. Alle regels met # bevatten informatie voor de compiler.
2. De namen tussen <... > zijn libraries van C.
3. Naam van de functie. Tussen de (...) is plaats voor argumenten. Void wordt later behandeld.
4. Openingsteken (delimiter) voor de statements die tot de functie behoren
5. Een statement. Dit statement plaatst een literal op het scherm:
   - printf is de naam van het statement
   - tussen haakjes het literal dat op het scherm moet komen. Een literal wordt altijd opgesloten tussen dubbele aanhalingstekens ("...")
   - Elk statement wordt afgesloten met een punt-komma (;)
6. Sluitteken (delimiter)

In dit voorbeeld bestaat het hele programma uit 1 functie (main) en 1 statement (printf). Het geheel wordt de **"Structuur van een programma"** genoemd.

Zo, de kop is er af. Je hebt nu kennis gemaakt met het eerste statement: **printf()**. Dit statement maakt het mogelijk data op het scherm te plaatsen.

Hoofd- en kleine letters

"C" is bijzonder gevoelig in het gebruik van hoofd- en kleine letters. Het is niet toevallig dat de statements van het voorbeeld in kleine letters zijn geschreven!

Let op KLEINE en GROTE letters bij statements!!!

PRINTF() is dus heel wat anders dan printf()!!!!!

Format en escape tekens

Wij gaan printf() wat nader bekijken:

```
main()
{
    printf("%s is %d miljoen km\n van de zon.", "Venus", 67);
}
```

Op het scherm zien wij:

Venus is 67 miljoen km
van de zon.

De **printf**() functie heeft het %s teken vervangen door de string "Venus" en het %d teken door het getal "67". Bovendien heeft er een sprong plaatsgevonden naar de volgende regel bij het teken "\n".

Nog een voorbeeld:

```
main()
{
printf("De letter %c wordt ", 'j');
printf("uitgesproken als %s .", "jee");
}
```

Op het scherm toont dat als:

De letter j wordt uitgesproken als jee.

Het teken: \ wordt de **Escape specifier** genoemd. Het % teken wordt de **Format specifier** genoemd. Bovendien zal de oplettende cursist zijn opgevallen dat er NIET naar een nieuwe regel gesprongen wordt zonder een toepasselijk escape teken, zelfs niet bij 2x het statement: printf().

Let ook even op de wijze waarop strings en characters worden vermeld; strings **altijd** tussen dubbele aanhalingstekens en characters **altijd** tussen enkele aanhalingstekens!

Variabelen en konstanten

In de vorige voorbeelden is gebruik gemaakt van konstanten (strings, characters en getallen) binnen de printf() functie. In plaats van konstanten mogen ook variabelen worden gebruikt. Deze variabelen dienen vooraf te worden **ge-declareerd** en **te worden assigned.**
Een declaratie bepaalt het soort gegevens dat in een variabele wordt geplaatst; een assignment vult een variabele met een waarde. De waarde kan een cijfer, getal, letter of string zijn. Declaraties MOETEN als eerste in een programma worden geplaatst.

Declaraties en assignments zijn ook "statements" en worden daarom afgesloten met een: ";".
Er bestaat nog een andere soort statements: kommentaar regels. Kommentaarregels mogen overal in een programma worden geplaatst in de vorm:

```
/* Dit is kommentaar */
```

Dan nu een voorbeeld:

```
main()                  /* Dit is een eenvoudig voorbeeld */
{
   int num;
   num = 2;
printf("Dit is nummer twee: %d", num);
}
```

Dit voorbeeld toont drie nieuwe elementen. Als eerste een kommentaar statement. Daarna het statement: printf(). Voordat dit wordt uitgevoerd, wordt eerst een variabele: "num" ge-declareerd als een heel getal ("int") en vervolgens wordt deze variabele de waarde: "2" gegeven. Dit laatste wordt **Assignment** genoemd.

In de functie printf() wordt de naam van de variabele gebruikt **zonder** aanhalingstekens. Zou het tussen aanhalingstekens staan dan beschouwt "C" het als een karakter of string.

Het teken: "=" is in dit geval geen vergelijkingsteken en behoort niet tot een rekenkundige bewerking maar is een onderdeel van de syntax bij het assignen van variabelen.

Elk statement afsluiten met een punt-komma!

Declaratie en assignment mogen ook in een keer worden geschreven:

```
main()
{
   int num = 2;
   printf("Dit is nummer twee: %d", num);
}
```

Nu een voorbeeld met karakters:

```
main()
{
   char ch1, ch2;
   ch1 = 'A';
   ch2 = 'Z';
   printf("Het alfabet loopt van: %c tot %c", ch1, ch2);
}
```

Een uitgebreider voorbeeld:

```
main()
{
   int ronde = 5;
   char heat = 'C';
   float tijd = 27.25;

   printf("De winnende tijd in heat %c", heat);
   printf("van ronde %d was %f.",ronde, tijd);
}
```

Op het scherm:

De winnende tijd in heat Cvan ronde 5 was 27.250000.

Twee printf() statements worden als 1 regel op het scherm gezet.

Let op spaties! Je kunt een of meer spaties inbouwen door tussen " en "van" een blanko positie in te voegen.

Format specifiers

Format Specifiers in het printf() statement worden gebruikt in kombinatie met de gewenste weergave van cijfers, getallen, karakters of strings.

De volgende **Format Specifiers** zijn mogelijk:

teken	omschrijving
%c	enkel karakter
%s	string van karakters
%d	heel decimaal cijfer / getal positief of negatief
%i	heel decimaal cijfer / getal positief of negatief
%f	floating point (decimaal getal)
%e	floating point (exponentieele getallen)
%u	heel decimaal getal zonder pos of neg teken
%x	hexadecimaal getal (b.v. 0F)
%o	hele octal getallen

De breedte van getallen kan ook worden ingesteld. Wij bekijken nogmaals het vorige voorbeeld:

```
main()
{
   int ronde = 5;
   char heat = "C";
   float tijd = 27.25;

   printf("De winnende tijd in heat %c", heat);
   printf(" van ronde %d was %.2f.",ronde, tijd);
}
```

Op het scherm: (nu wel met spatie)

De winnende tijd in heat C van ronde 5 was 27.25.

Op deze wijze is het aantal posities achter de punt afgebroken op 2. Dit kan ook worden ingesteld voor het totale getal:

```
printf("De leeftijd is %2d.",leeftijd);
```

Nu wordt de leeftijd weergegeven in maximaal 2 cijfers. (en wat nu als de persoon ouder is dan 100 jaar?!!)

Kombinaties zijn ook mogelijk:

```
printf("%8.1f%8.1f%8.1f\n", 3.0, 12.5, 523.3);
printf("%8.1f%8.1f%8.1f\n", 300.0, 1200.5, 5300.3);
```

Dat ziet er op het scherm zo uit:

```
3.0     12.5     523.3
300.0   1200.5   5300.3
```

met overal 8 posities voor de getallen en 1 positie achter de komma. Er wordt niet afgerond!

Nog enkele mogelijkheden tot besluit:

```
char c1, c2, c3;
c1 = 'a';
c2 = 97;
c3 = Ox61;
```

- c1 = = > normaal character
- c2 = = > plaats van character in de ASCII tabel
- c3 = = > hexadecimaal

Escape sequences

U hebt al kennis gemaakt met een **escape character: "\n"**. Hieronder volgt nog een stel:

teken	omschrijving
\n	nieuwe regel
\t	tab karakter
\b	backspace
\r	carriage return
\f	formfeed
\'	enkel aanhalingsteken
\"	dubbel aanhalingsteken
\\	backslash
xdd	ASCII code in hex representatie (elke d is een hex teken)
ddd	ASCII code in octal notatie (idem)

Een voorbeeld van "\xdd":

```
main()
{
    printf("Hier is een karakter: \xDB");
}
```

Dat ziet er op het scherm uit als een vierkant vlakje. (kan ik niet weergeven op papier - sorry)

Een open vierkantje op het scherm zou zo gemaakt kunnen worden: (kan ik ook niet op papier weergegeven - helaas)

```
main()
{
    printf("\xC9\xCD\xBB\n");
    printf("\xC8\xCD\xBC\n");
}
```

Probeer dit later maar eens uit als oefening. Raadpleeg een ASCII tabel voor de betekenis van de diverse hex tekens.

Functie scanf()

Tijd voor een nieuwe functie. Deze functie heet: scanf() en lijkt op functie: printf(). Hij doet echter het omgekeerde van printf(); deze functie leest iets van het toetsenbord. Een voorbeeld:

```
main()
{
float jaren, dagen;
printf("Geef uw leeftijd in jaren: ");
scanf("%f", &jaren);
dagen = jaren * 365;
printf("U bent %.1f dagen oud.\n", dagen);
}
```

De eerste regels kunnen geen problemen meer geven. De regel met **scanf()** plaatst de input van het toetsenbord in variabele: "jaren". In dit geval **MOET** de naam van de variabele beginnen met een "&". Dit teken ("&") wordt de **"Address Operator"** genoemd en zorgt voor een verwijzing naar het adres in het geheugen waar de variabele staat. Dit heeft te maken met pointers en zal later in deze cursus summier worden behandeld.

Daarna ziet u een berekening en tenslotte wordt de uitkomst op het scherm geplaatst met behulp van de functie: printf().

Op het scherm ziet de hele operatie er zo uit:(stel, het programma heet: "leeftijd")

```
C> Leeftijd
Geef uw leeftijd in jaren: 2
U bent 730.0 dagen oud.
```

Of:

```
C> Leeftijd
Geef uw leeftijd in jaren: 48.5
U bent 17702.5 dagen oud.
```

- Format specifiers zijn hetzelfde als bij printf().
- Variabele namen beginnen met "&"
- Variabelen moeten vooraf ge-declareerd te zijn.

String in scanf()

Tot nu toe hebben wij in de declaratie van variabelen met datatype: "char" variabelen ge-declareerd waarin slechts plaats was voor 1 character. Nu toon ik u hoe een string kan worden ge-declareerd:

```
main()
{
   char naam[20];

    printf("Enter uw naam: ");
   scanf("%s", naam);
   printf("Mijn naam is: %s", naam);
}
```

LET OP! Let op het verschil met de variabele naam: "naam" in de functie: scanf(); het wordt NIET vooraf gegaan door een "&"! Dat is goed en houdt ook verband met pointers. Onthoudt het verschil met een declaratie van 1 character; bij een string geen "&" gebruiken. In een volgend deel kom ik daarop terug.

Hoe maken wij een programma?

Dit is een moeilijk onderwerp! Dat komt omdat elk "C" produkt weer andere methoden en regels kent. Er wordt wereldwijd gewerkt aan een normalisatie maar daarop kunnen wij niet wachten.

Ik zal eerst wat algemeenheden beschrijven. Daarna of daarnaast moet ieder de voorschriften van het aangeschafte pakket nauwkeurig uitvoeren.

De meeste "C" compilers werken met een editor waarmee een source in "C" geschreven kan worden. Heeft uw pakket geen editor dan gebruikt u een editor waarmee GEEN verborgen informatie wordt gemaakt. WordPerfect is berucht! Nadat de source is geschreven moet voor sommige "C" pakketten nog een z.g. MAKE file worden gemaakt. Andere pakketten doen dit automatisch en weer andere compilers hebben geen MAKE file nodig. Hier valt dus nog het een en ander uit te zoeken...

Een MAKE file bevat de naam van de source file en een aantal opties die nodig zijn voor de compiler en de linker. Bij de MAKE file hoort een programma dat, met de MAKE file en de source file als input, automatisch uw programma compileert en linked. Meestal heet dit programma NMAKE.

- Volg zorgvuldig de aanwijzingen bij het "C" pakket voor installatie, compilatie, linken, de editor en eventueel de MAKE file.
- De naam van een source file is vrij; de tweede qualifier is altijd: "C".
- Let op hoofd- kleine letters. "C" is hiervoor gevoelig
- Schrijf de programma"s ge-struktureerd; dat wil zeggen netjes inspringen om op die manier een goed, overzichtelijk en duidelijk programma te verkrijgen. Plaats kommentaar in de programma"s. Na korte tijd weet je al niet meer wat je bedoeld had...
- Als alles goed gaat - zonder syntax fouten - krijg je een ---.EXE file welke uitgevoerd kan worden. Tik simpel de naam van het programma in om het uit te voeren.

O ja, er is nog iets. In het eerste voorbeeld zie je twee include statements en de toevoeging "void" in het main statement. Bij de andere voorbeelden heb ik dat weggelaten.

De meeste "C" compilers bedenken zelf deze toevoegingen als deze ontbreken. Dat worden **DEFAULTS** genoemd. De compiler zal dan boodschappen geven waaruit blijkt dat "hij" ontdekt heeft dat de maker van het programma de includes en voids heeft weggelaten maar zal het programma toch goed compileren. Mocht het echter keihard stuk lopen, schrijf de programma's dan zoals in het eerste voorbeeld is weergegeven.

Tot slot nog dit: BORLAND C++ (prijs f. 1800,--) is Objectgeoriënteerd. Als je daarvan geen kaas hebt gegeten kun je dit pakket maar beter vergeten. Dan is een betere keuze MICROSOFT C 6.00.

Oefeningen:

1. Begin met het intikken van enkele voorbeelden. Compileer en voer de voorbeelden uit.

2. Schrijf een programma dat op het scherm displayed:

   Mr. Green = 42
   Mr. Brown = 58.

3. Ontwikkel een programma dat uw naam vraagt en op het scherm plaatst.

   Voor rekenkundige bewerkingen worden de gangbare operators gebruikt zoals + - * / voor respectievelijk optellen, verminderen, vermenigvuldigen en delen.

4. Schrijf nu een programma dat graden Fahrenheit omzet in graden Celsius. (U tikt bij uitvoering van het programma het aantal graden Fahrenheit in en het programma antwoordt met het aantal graden Celsius)

Literatuuroverzicht

1. De kleine C gids, Academic service, ISBN 90 6233 467 9
2. Programmeren met C, idem, ISBN 90 6233 657 4
3. Microsoft C, Howard W.Sams comp, ISBN 0 672 22661 8
4. Ref manual C, IBM, internal

Hans van Boheemen

Ik heb interesse in de KGN en wil

☐ Lid worden van de KGN ☐ Meer informatie over de KGN

Naam : _____

Adres : _____

Postcode en Woonplaats : _____ _____

Datum : _____ Handtekening : _____

Dit strookje kunt u ingevuld opsturen aan het secretariaat van: KIM Gebruikersclub Nederland
Postbus 1336
7500 BH Enschede

Teletekst (deel 3)

Inleiding

In deze aflevering wordt de bij de teletekst-hardware behorende software beschreven. De software zelf kan gedownload worden vanaf het club-bulletin board The Ultimate of verkregen worden op de clubbijeenkomsten.

Het teletext programma TT.EXE is voor de ontvangst, het afbeelden en tijdelijke opslaan op disk van teletext pagina's. Het gehele programma is vrijelijk te gebruiken, te copieren en aan te passen, echter alleen voor hobby-doeleinden.

Commercieel gebruik (geheel of gedeeltelijk) of verspreiding is alleen in overleg met de auteur toegestaan.

Enkele eigenschappen

- TT kan totaal 50 pagina's in het geheugen opslaan.
- TT kent 20 zgn. Catalogs welke een onbeperkt aantal subpagina's kunnen bevatten. (uiteraard max. 50 pagina's met 1 Catalog).
- TT kent de zgn. OFFLINE MODE waarby pagina's van disk kunnen worden geladen wanneer de IIC line afwezig is.

```
*** Teletext V1.0B ***

Page : 100 Catalog : 00

 <A> Ascii conversionoff
 <T> Tune/Test channel
 <C> Clear all catalogs
 <P> Print catalog
 <L> Load catalog from disk
<S><W> Save/Write page to disk
ENTER Search for page(s)
 CUP/CDW Select subpage up/down
 PUP/PDW Select catalog up/down
 HOM/END Select catalog home/end

 <SP> Stop Search
 <Q> Quit to DOS

No. Subpages 00 Read pages 00

Status : xxxxxxxxxxx
```

Fig. 1: het menublok

Het menublok

Commando's: (Zie het menublok in figuur 1.)

Page/Enter: Het page nummer (100..899) kan ingetoetst worden en met een Enter worden bevestigd. Het page nummer kan met de cursor <- en -> toetsen worden doorlopen. TT zoekt bij het opgeven van een pagina ook pagina +1,+2 en +3 op die bij de volgende aanvraag op het scherm kunnen worden getoond.

Het ingegeven nummer kan tevens bestaan uit wildcards '*' waarna bij het zoeken alle pagina's (voorzover de buffers dit toelaten) worden ingelezen. Afhankelijk van het gebruikte teletext systeem worden in de cyclustijd misschien niet alle pagina's ingelezen.

Catalog: Dit geeft het momentele Catalogus nummer weer.

A: Toggle Ascii conversion on/off. Wanneer Ascii conversion aan is worden alle niet ascii tekens uit de pagina's gefilterd.

T: Test mode en tuning channel (zie menu).

C: Wis alle catalogs/pagina's.

P: Print de momentele catalog in een formaat van 9 pagina's op één blad. (PRN1)

L: Laad een catalog van disk. (extension .ttd)

S: Save een pagina naar disk.

W: Write een catalog naar disk.

CUP/CDW: Selecteer van een catalog de subpagina's.

PUP/PWD: Selecteer een catalog.

HOM/END: Selecteer eerste en laatste catalog.

SP: Spatiebalk is break search.

Q: Verlaat TT.

Subpages: Het aantal verwachte subpagina,s van een catalog. Het aantal verwachte subpagina's wordt gelezen uit de eerste gelezen pagina. Als dit '??' is wordt er zolang gelezen tot een reeds gelezen subpagina nog een keer wordt gelezen.

Read Pages: Het aantal gelezen pagina's.

Status: Status kent een aantal status- en foutmeldingen.

Het paginablok

Links van het menublok op het scherm verschijnt het paginablok.

Fig. 2: voorbeeld van een paginablok

De betekenis van de velden wordt hierbij kort beschreven.

M0:	Geheugen nummer van de CCT chip (0..3).
BRTN...	De standaard koptekst van een teletext pagina met paginanummer,zenderinfo,tijd en datum.
nieuws...	De inhoud van de pagina.
Subpage:	Het subpage nummer van deze pagina.
Status:	De status van deze pagina kan bestaan uit de volgende letters: CMTRIS.
	Deze letters komen overeen met een bit uit de status data uit de pagina. De betekenis is als volgt:
	C bit C4:Wis pagina, nieuwe data.
	M bit C5:Nieuwsinformatie.
	T bit C6:Ondertitelings pagina.
	R bit C8:Vernieuwde tekst van deze pagina.
	I bit C9: Pagina buiten normale cyclus (meestal index).
	S bit C11:Pagina met doorrollende koppen.
Noot:	De verschillende omroepstations maken m.i. niet consequent gebruik van deze codes.

Command line parameters

Ten behoeve van batchbewerkingen kunnen pagina's worden gezocht en opgeslagen op disk vanaf de command line. Tijdens het zoeken blijft het scherm leeg. Mocht er bij het zoeken een probleem zijn dan wordt de output file niet aangemaakt.

De parameters zijn: (in willekeurige volgorde)

/a of /A	Ascii conversie aan (default Uit)
/p100 of /P100	Pagina nummer (met alle subpagina's) (default 100)
/f11 of /F11	Channel nummer TV
tt.ttd	Filenaam (zonder '/') (default tt.ttd)

Mocht één van de parameters niet worden opgegeven dan wordt de default gebruikt.

Voorbeeld:	tt /p300 /a /f12 koers
of	tt /a koersen /p400

TTGET, TTCONV en TTSHOW

TTGET.EXE:

ttget.exe is voor de abstractie van data uit tt-files. Deze tool is geschikt om b.v. koersen op te slaan in een log-file. De output van dit programma gaat naar standard output en het is dus mogelijk door middel van redirection (> of >>) de output naar een file te schrijven cq aan een file toe te voegen.

De foutmeldingen gaan naar de console.

De parameters zijn: (in willekeurige volgorde)

/s of /S	Subpage nummer 1 tot ..
/l of /L	Linenummer 1 tot 24
/c of /C	Column nummer 1 tot 40
/n of /N	Lengte uitvoer 1 tot ..
/x of /X	Output een CR/LF na uitvoer (Optioneel)
xxxx.xxx	Filenaam (zonder '/')

Mocht één van de parameters niet worden opgegeven dan volgt foutmelding.

Voorbeeld: ttget /s2 /l5 /c8 /n10 /x p400.ttd

TTCONV.EXE:
ttconv.exe converteert tt-files tussen tt en ttedit en v.v.
Voorbeeld: ttconv 100 p100.ttd
of ttvonv p566.ttd 124

TTSHOW.EXE:
ttshow.exe is voor het weergeven van ttedit-files op de tv t.b.v. videobewerking (titels e.d.). Het programma kan 18 ttedit-files laden. Er zijn scrolling en beeldmanipulatie mogelijkheden.

(zie Menu van het programma)

Mogelijke toekomstige aanpassingen.

In het volgende overzicht staan enkele ideeën voor mogelijke toekomstige uitbreidingen.

- Het uitbreiden van het aantal pagina's.
- Tijdens zoeken kopregel tonen.
- Ghostpagina's bewerken.
- Tijdsgeplande aanvraag pagina's.
- Gebruiker suggesties.

Momentele tekortkomingen/bugs

Bij het tonen van de filenamen tydens het invoeren van een input- of outputfilenaam wordt de momentele directory verkeerd weergegeven. ('\' teken) Dit wordt veroorzaakt door het geladen teletext font.

T. Bijkerk, Hengelo

Voortgang KGN68k deel 12

Technische voortgang is er deze keer niet veel te melden. Wel andere vooruitgang.

Print

Van Dombo68k heb ik nog steeds geen printplaat van laten maken. Pogingen om het beter te willen doen als de vorige keer hebben tot heden nog geen resultaat opgeleverd.

Software

De GNU omgeving is een prachtige omgeving om software te ontwikkelen. Alles wat een programmeur nodig heeft is aanwezig. Dus ook een assembler. De source voor die assembler moet echter aan de GNU syntax voldoen. Dat is helaas een ander formaat als Motorola gebruikt in zijn documentatie gebruikt. Nu worden we dus beloond dat we de KGN68k assembler zelf geschreven hebben. Hierdoor kunnen we makkelijk een toevoeging maken voor een output formaat dat geschikt voor de GNU omgeving. Wat ons echter weer tegen zit is dat de grote man achter de KGN68k assembler veel bezig is met ons clubblad de μP Kenner.

De grote man achter de KGN68k assembler is veel bezig met ons clubblad.

Tektronix

Als we Combo68k een andere clock geven als de processorkaart loopt het systeem vast. We draaien nu dan ook maar met één clockgenarator. Het probleem hebben we laten zitten voor als we een keer de beschikken hebben over een Logic Analyzer. Nu hebben wij de firma Tektronix bereid gevonden dat wij gebruik mogen maken hun meetapparatuur waar onder een Logic Analyzer. Hoe het bezoek aan Tektronix in Hoofddorp verlopen is kan ik de volgende keer pas vertellen. De kontakten zijn nog maar vers. En voor we op bezoek gaan moeten we eerst weer het probleem introduceren.

Geert Stappers (04781-41279)

De Motorola 68030; het hart van KGN68k (deel 5)

Inleiding

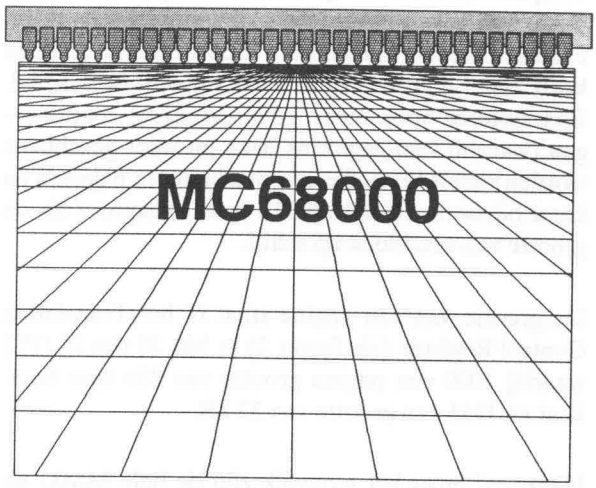
In deze aflevering wordt de behandeling van de 68030-processor (voorlopig) afgerond met de behandeling van het laatste (en naar mijn mening meest spectaculaire) onderdeel van de processor: de Memory Management Unit of MMU. Dat de behandeling van de processor stopt wil niet zeggen dat de serie ook stopt, welnee, er zijn nog een heleboel zaken te beschrijven, maar daarover straks meer.

Het programmeermodel van de MMU

Om maar meteen met de deur in huis te vallen, is in figuur 1 het programmeermodel van de MMU weergegeven. We zien in deze figuur twee registers met een breedte van 64 bits; de zogenaamde Root Pointers, drie registers met een breedte van 32 bits en een status register met een breedte van 16 bits.

Uiteraard zullen in dit artikel deze registers uitgebreid aan bod komen.

Zoals uit de vorige afleveringen wel bekend is, bestaat er een onderscheid tussen de logische adressen die in het programma gebruikt worden en de fysieke adressen waarop we het geheugen vinden. Tussen deze twee adressen bestaat er een vertaaltabel met behulp waarvan de logische adressen omgerekend worden naar de fysieke adressen. Voor de beschrijving van de MMU ga ik er steeds van uit dat het gezochte logische adres altijd afgebeeld wordt op een fysiek adres en dat de gezochte informatie dus "ergens" in het fysieke geheugen staat. Mocht dit niet zo zijn, dan zal de processor een zogenaamde Bus Error Exception uitvoeren waarna het Operating System ervoor zorgt dat het gezochte logische adres inderdaad in het fysieke geheugen terecht komt.



Het Translation Control Register

Omdat het natuurlijk ondoenlijk is voor elk afzonderlijk logisch adres een entry in de vertaaltabel te hebben (de vertaaltabel zou dan twee keer zo groot worden als het adresseerbare geheugen), zijn het logische en het fysieke geheugen opgedeeld in "pagina's" of "pages". In de vertaaltabel staat dan dat logische pagina 1 bijvoorbeeld afgebeeld wordt op fysieke pagina $100 etc. De MMU kent 8 verschillende pagina groottes variërend van 256 byte tot 32 kB. Verder is de grootte van een pagina altijd een macht van 2 zodat in een logisch adres van 32 bits een aantal bits de logische pagina aangeven en de rest van de bits het adres van de geheugenlokatie binnen de pagina. Hoe groot men een pagina kiest is afhankelijk van een groot aantal factoren. Deelt men het geheugen op in pagina's van 32 kB, dan is de vertaaltabel kort maar wordt het geheugen opgedeeld in grote brokken die aan de diverse processen (zeg maar programma's in het geheugen) toegewezen kunnen worden. Vaak houdt dit in dat grote stukken geheugen "verspild" worden. Kleine pagina's maken de verspilling minimaal maar zorgen er voor dat de

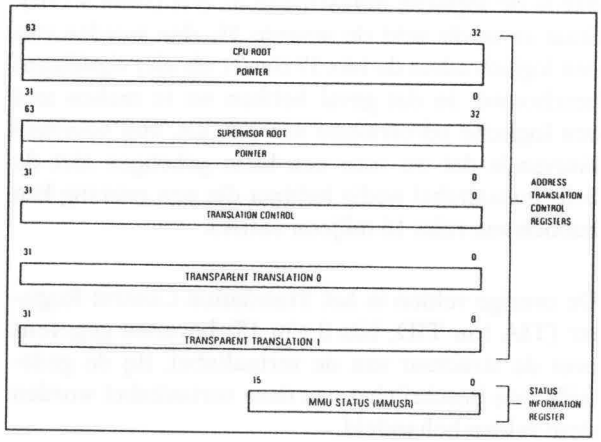


Fig. 1: programmeermodel van de MMU

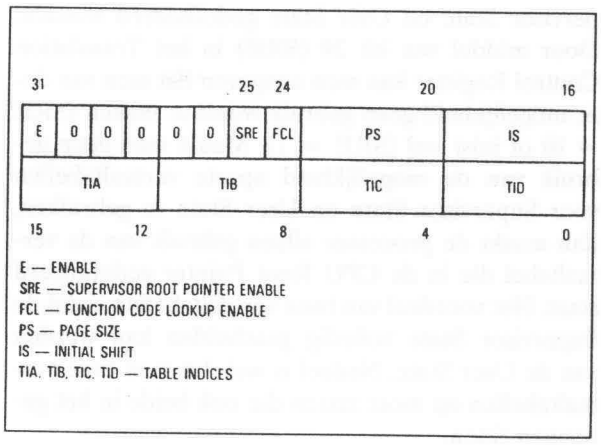


Fig. 2: het translation control register

vertaaltabel erg groot (en daarmee langzaam) wordt. In de 68030 User Manual staat als voorbeeld een computersysteem dat speciaal ontworpen is voor zogenaamde reken-intensieve toepassingen (een zogenaamde Number Cruncher). Een dergelijk systeem heeft baat bij grote pagina's van bijvoorbeeld 32 kB. Een systeem voor meer administratieve toepassingen (waarbij veel gegevens heen en weer geschoven worden) heeft behoefte aan wat kleinere pagina's en komt bijvoorbeeld uit op 512 byte, evengroot als de grootte van een block op schijf.

De grootte van een pagina staat in het Translation Control Register (zie figuur 2) in bits 20 t/m 23 (PS) waarbij 1000 een pagina grootte van 256 byte betekent en 1111 een grootte van 32 kB.

Uiteraard moet het mogelijk zijn de hele MMU uit te schakelen. Tenslotte zul je na een Reset op zijn minst de vertaaltabel moeten gaan vullen. De aan en uit schakelaar van de MMU is bit 31 (E) van het Translation Control Register. Is dit veld gevuld met een 0, dan is de MMU uitgeschakeld en zijn de fysieke adressen precies gelijk aan de logische adressen. Een waarde 1 in dit veld schakelt de MMU in waarna de fysieke adressen onder het beheer van de vertaaltabel vallen. Behalve via het Translation Control Register kun je de MMU ook met een extern signaal uitschakelen. Dit is het MMUDIS-signaal dat gebruikt kan worden voor Emulators, test- en andere doeleinden.

Eén datagebied lijkt me niet zinvol; drie gebruikers die tezamen één programma wijzigen?

Het startpunt van de vertaaltabel en nog wat extra informatie staat in een register in de MMU. Hiervoor zijn twee 64 bits registers aanwezig: de CPU Root Pointer en de Supervisor Root Pointer. Hiermee kunnen verschillende vertaaltabellen voor Supervisor State en User State gedefinieerd worden. Door middel van bit 24 (SRE) in het Translation Control Register kan men aangeven dat men van deze mogelijkheid geen gebruik wenst te maken (SRE = 0) of juist wel (SRE = 1). Maakt men geen gebruik van de mogelijkheid aparte vertaaltabellen voor Supervisor State en User State te gebruiken, dan maakt de processor alleen gebruik van de vertaaltabel die in de CPU Root Pointer gedefinieerd staat. Het voordeel van twee vertaaltabellen is dat de Supervisor State volledig gescheiden kan worden van de User State. Nadeel is wel dat men twee vertaaltabellen op moet zetten die ook beide in het geheugen zitten.

Bit 24 (FCL) is ook zo'n schakelaar. Het geeft aan af de functiecodes onderdeel uitmaken van het logische adres dat vertaald wordt. Dit betekent dat men met dit bit kan kiezen of een instructie op het logische adres $12342 in het fysieke geheugen volgt op de data op logisch adres $12340 of dat instructies en data op volstrekt verschillende fysieke adressen liggen, ook als hun logisch adres opéénvolgend is. Hetzelfde geldt ook voor het onderscheid Supervisor State/User State die ook af te lezen valt aan de functiecodes. Het grote voordeel van het maken van onderscheid tussen instructies en data vindt je in een Multi Tasking omgeving. In dat geval zijn er meerdere taken (programma's) actief in het geheugen van de computer. De processor werkt een tijdje (in jargon een time slice) aan de ene taak, krijgt een interrupt en gaat vervolgens werken aan een andere taak in het geheugen. Nu kan het natuurlijk heel goed voorkomen dat meerdere taken dezelfde code gebruiken (drie gebruikers zijn gelijktijdig aan het editten met VI). In principe heb je nu maar één maal de code van dat programma nodig en voor elke taak een eigen data-gebied (één datagebied lijkt me niet zinvol; drie gebruikers die tezamen één programma wijzigen?). Nu zijn er, zeker voor de 68000-familie, meerdere technieken waarmee je ervoor kunt zorgen dat je gebruik maakt van "Shared Code", maar één ervan is het volledig door het Operating System te laten regelen met behulp van de MMU.

Omdat de grootte van de vertaaltabel niet alleen afhangt van de grootte van een pagina maar ook van het totale (logische) adresbereik, kan men in het Translation Control Register, in bit 16 t/m 19 (IS) aangeven uit hoeveel significante adresbits een logisch adres bestaat. Is IS gevuld met de waarde 0, dan is de logische adresruimte 32 bits (ruim 4 GB). Staat er in dit veld de waarde 15, dan worden van een logisch adres de bits 17 t/m 31 als niet significant beschouwd. In dat geval hebben we te maken met een logische adresruimte van 128 kB. Het voordeel hiervan is dat we voor een klein geheugen niet de hele vertaaltabel nodig hebben die een omvang kan hebben van ruim 16 miljoen entries.

De overige velden in het Translation Control Register (TIA t/m TID, bits 0 t/m 15) bevatten gegevens over de structuur van de vertaaltabel. Bij de gedetailleerde beschrijving van deze vertaaltabel worden deze velden behandeld.

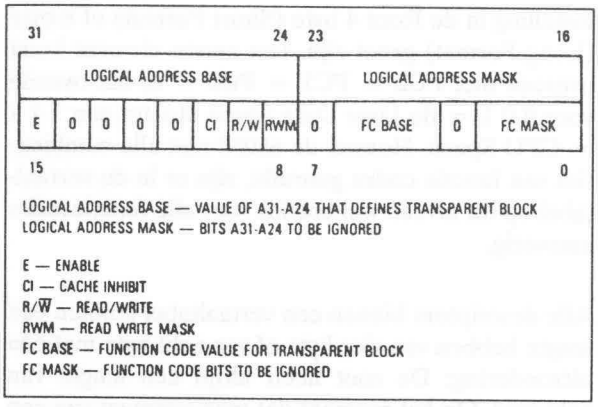


Fig. 3: structuur van een transparent translation register

De Transparent Translation Registers

Ook als de MMU niet uitgeschakeld is, kan men toch in de geheugenruimte van de processor maximaal twee gebieden definiëren waarvoor er geen omrekening van een logisch adres naar een fysiek adres plaatsvindt. Hier zijn de logische en fysieke adressen dus identiek.

Om deze blokken te kunnen vastleggen bezit de 68030 twee zogenaamde Transparant Translation Registers, TC0 en TC1 waarvan de structuur in figuur 3 is weergegeven. In de eerste plaats is er weer een Enable bit (bit 15) dat aangeeft of het betreffende Translation Control Register gebruikt wordt. Verder vinden we een bit (Cache Inhibit, 10) dat aangeeft of de informatie op het betreffende adres "Cacheable"; dus of die informatie overgenomen mag worden in de instructie- of data-cache. Indien dit Cache Inhibit bit 1 is, zal de informatie binnen dit adresbereik niet overgenomen worden in de caches, sterker nog, de processor zal door middel van het CIOUT-signaal ook naar eventuele externe caches aangeven dat de informatie niet in een cache mag worden opgenomen. Aangezien men bijvoorbeeld periferie-bouwstenen in dit adresbereik opneemt is dit een zeer nuttige eigenschap.

Door middel van het R/W bit (bit 9), in combinatie met het RWM bit (bit 8) kan worden aangegeven of zowel een lees als een schrijf-benadering (RWM = 1) of alleen de Lees- (RWM = 0, R/W = 1) resp. de schrijfbenadering van het geheugen (RWM = 0, R/W = 0) buiten de vertaaltabel om gaat. Iets dergelijks geldt ook voor de functiecodes. Zoals bekend, kan men de functiecode bits opvatten als een uitbreiding van het logische adres met drie bits. Aan de hand van deze functiecodes weet je precies waarom de processor het geheugen benadert. Door middel van de FC MASK (bit 4 t/m 6) kan men aangeven welke functie-code bits buiten beschouwing gelaten worden bij de bepaling van het transparante adresbereik. Zijn alle bits in dit masker 0, dan zal het transparante bereik alleen gelden voor precies die functiecode die in de FC BASE (bit 0 t/m 2) staat. Is bit 6 van het Transparant Translation Register een 1, dan heeft dit tot gevolg dat een ingesteld adresbereik transparant voor zowel de User State van de processor als de Supervisor State omdat FC2 het onderscheid tussen deze twee toestanden aangeeft.

Blijft over de Logical Adress Base en de Logical Adress Mask. De Base geeft aan voor welke waarden van adresbits 24 t/m 31 de omrekening transparant is. Dit betekent dat een transparant blok altijd minimaal 16 MB groot is. Door het setten van bits in de Logical Adress Mask kan men dit bereik nog vergroten. Elk bit dat in dit masker geset wordt geeft aan dat het bijbehorende bit in de Base een zogenaamde don't care is. Op deze manier kan men dus grotere transparante blokken maken of meerdere blokken van 16 MB of groter.

Tenslotte wordt nog opgemerkt dat de transparante blokken in beide Transparant Control Registers elkaar geheel of gedeeltelijk mogen overlappen en dat adressen in het CPU-gebied (functiecode 7) altijd transparant zijn. Dit laatste geldt overigens in het algemeen: adressen met FC0 t/m FC2 = 1 (CPU-gebied) worden altijd buiten de MMU om benaderd.

Opbouw van de vertaaltabel

De vertaaltabel tussen logische en fysieke adressen wordt in de praktijk opgebouwd door het Operating System. Deze opbouw zal dus binnen het Operating System geprogrammeerd moeten worden. De processor is zelf in staat een vertaaltabel te doorzoeken om zodoende de omrekening van een fysiek naar logisch adres te kunnen vinden. De stappen en geheugen-benaderingen die hiervoor nodig zijn staan in een programmaatje die in de processor, in zogenaamde microcode, zit. Uiteraard moet de vertaaltabel dan wel zodanig opgebouwd zijn dat de processor deze tabel kan lezen.

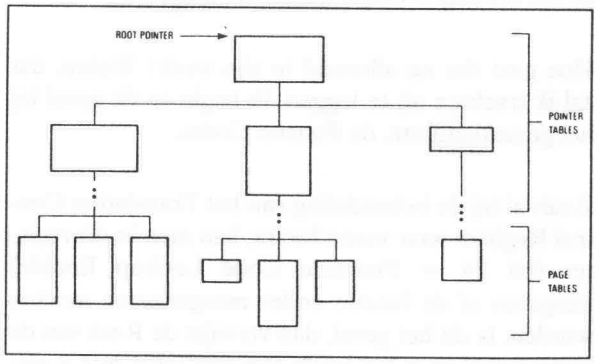


Fig. 4: schematische weergave van een vertaaltabel

De algemene opbouw van de vertaaltabel is een boomstructuur. In figuur 4 is een dergelijke structuur getekend. In de literatuur wordt voor de vertaaltabel de term Translation Table gebruikt. Verder noemt men het enkele element boven in de figuur de "Root" of "Wortel". De wortel van een boom wordt in de informatica altijd bovenaan getekend zodat bomen hier altijd naar beneden groeien. Om voor een lopend programma de vertaaltabel te kunnen gebruiken, moet de root opgeslagen worden in één van de twee Root Pointers in de MMU.

De boom die de vertaaltabel voorstelt is opgebouwd uit zogenaamde "Descriptors". Hiervan zijn er voor de 68030 elf verschillende types die ik niet stuk voor stuk zal behandelen. Ik moet mij noodgedwongen beperken tot de belangrijkste. De takken van de boom worden gevormd door de zogenaamde Table Descriptors. Dit zijn data-structuren die weer verwijzen naar data-structuren op een lager niveau. In figuur 4 zijn deze descriptors aangegeven als "Pointer Tables". De bladen van de boom worden gevormd door de Page Descriptors (Page Tables in de figuur) die de informatie bevatten waarmee een logisch adres omgezet wordt naar een fysiek adres. Hoewel figuur 4 behalve de Root slechts één niveau van Table Descriptors bezit, kan men de 68030 zo programmeren dat hij de volgende boom kan gebruiken:

- De Root (Root Pointer Descriptor)
- Een Functie Code niveau
- Drie niveaus met Table Descriptors
- Eén niveau met Page Descriptors

Totaal dus vijf niveaus plus één niveau voor de Functie Codes. Gelukkig hoeven al die niveaus niet gebruikt te worden. Het minimum is één niveau waarbij de Root een zogenaamde Early Termination Descriptor bevat. Verder is alles tussen dit minimum en het bovengenoemde maximum mogelijk.

Hoe gaat dat nu allemaal in zijn werk? Welnu, dat zal ik trachten uit te leggen. Ik begin in dit geval bij het gemakkelijkste, de Functie Codes.

Zoals al bij de behandeling van het Translation Control Register naar voren kwam, kan men in dit register (bit 24 = Function Code Lookup Enable) aangeven of de functie codes meegenomen moeten worden. Is dit het geval, dan verwijst de Root van de vertaaltabel naar het begin van een tabel met 8 elementen. Deze elementen kunnen, afhankelijk van de instelling in de Root 4 byte (Short Format) of 8 byte (Long Format) groot zijn. Het eerste element komt overeen met FC2 = FC1 = FC0 = 0, het tweede voor 0 0 1 = de User Data Space etc. tot aan 1 1 1 = CPU Space. Hoewel de 68030 niet alle combinaties van functie codes gebruikt, zijn er in de vertaaltabel op dit niveau wel entries voor alle functiecodes aanwezig.

Alle descriptors binnen een vertaaltabel kunnen een lengte hebben van vier byte of van acht byte met één uitzondering: De root heeft altijd een lengte van acht byte. Op het moment dat men overgaat van een acht byte descriptor naar een vier byte descriptor wordt dit in de descriptor die naar deze vier byte descriptor verwijst aangegeven. Hetzelfde geldt voor een overgang van vier byte naar acht byte.

Hiervan zijn er voor de 68030 elf verschillende types die ik niet stuk voor stuk zal behandelen.

In het algemeen zal na de functie codes, of meteen na de root als de functie codes niet gebruikt worden, nog minimaal 1 niveau komen. Hier komen de velden TIA, TIB, TIC en TID uit het Translation Control Register om de hoek kijken. In het veld TIA wordt aangegeven hoeveel bits van het logische adres als index in de tabel op dit eerste niveau fungeren. Aangezien de velden een breedte hebben van 4 bits kan men dus maximaal 15 bits als index gebruiken voor maximaal 32 k elementen. Hetzelfde geldt in principe voor het derde (TIB), vierde (TIC) en vijfde (TID) niveau. Aangezien een logisch adres 32 bits bevat heeft het uiteraard geen zin voor alle niveaus 15 bits te decoderen, nee de berekening gaat als volgt:

- Het veld IS in het Translation Control Register geeft aan hoeveel bits, gerekend vanaf bit 31 niet mee doen
- Het veld TIA geeft aan hoeveel bits, gerekend vanaf het laagste bit dat niet mee doet volgens IS worden gebruikt voor de opsplitsing op het eerste niveau
- Evenzo voor TIB waarbij men rekent vanaf het laagste bit uit TIA, TIC en TID.
- De Page Size (PS) geeft tenslotte aan welke bits uit het logische adres gebruikt worden om binnen de fysieke pagina de geheugenlokatie te adresseren.

Aangezien een logisch adres uit 32 bits bestaat (0 t/m 31) moet dus altijd gelden:

IS + TIA + TIB + TIC + PS = 32

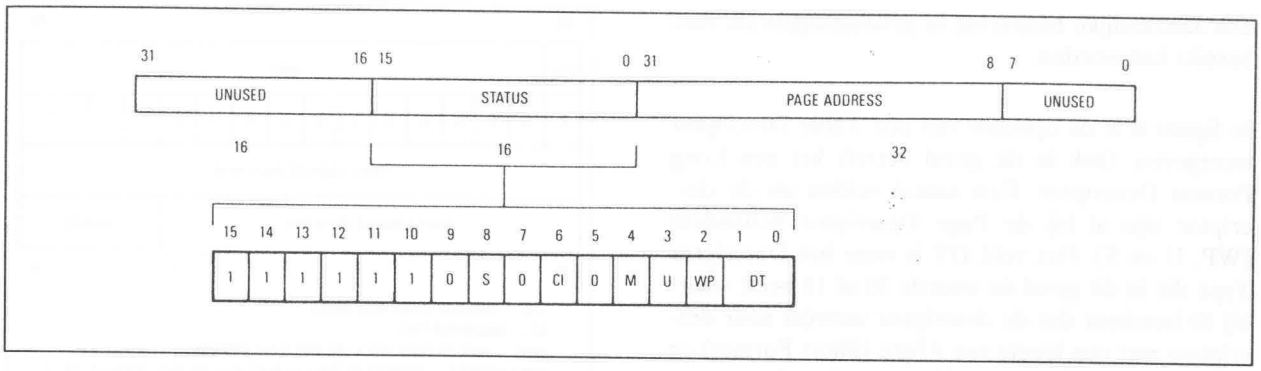


Fig. 5: long format page descriptor

Waarbij opgemerkt wordt dat als TIB de waarde nul bevat de inhoud van TIC en TID ook beschouwd worden als zijnde 0. Evenzo voor TIC waarbij TID ook niet mee telt. Vult men bijvoorbeeld TIB met 0, dan heeft men slechts twee niveaus in de vertaal tabel: de Root en een niveau van Page Descriptors op niveau "A".

Om te laten zien wat je allemaal met de MMU kunt, zal ik twee vormen van descriptors wat uitgebreider beschrijven. Hierbij begin ik met de Long Format Page Descriptor uit figuur 5. In de eerste plaats zien we het zogenaamde Page Adress in bits 8 t/m 31. Voor een pagina grootte van 256 byte vormen deze bits plus de laagste 8 bits uit he logische adres het fysieke adres. Voor grotere pagina's worden extra bits niet gebruikt. Het aantal bits van het Page Address dat niet gebruikt wordt is de waarde van PS minus acht.

Het veld DT geeft het descriptor type aan en is in dit geval 01 wat betekent een Page Descriptor. Het WP (Write Protect) bit geeft aan of er in de pagina alleen gelezen of ook geschreven mag worden. Het U (Used) bit in het M (Modified) bit worden door de processor gezet. Als het Operating System deze bits op 0 initialiseert, dan zal de processor U de waarde 1 geven als hij deze descriptor (en dus de bijbehorende pagina) benadert heeft. Was deze benadering voor een schrijfoperatie, dan zal bovendien het M bit de waarde 1 krijgen. Op deze manier kan het Operating System constateren welke pagina's in het geheugen gewijzigd zijn. Het CI (Cache Inhibit) bit geeft aan of informatie van de pagina overgenomen mag worden in de caches. Is dit niet het geval, dan zal dat bovendien door middel van het externe signaal CIOUT aangegeven worden. Het S (Supervisor Only) bit geeft aan dat de pagina alleen in Supervisor State benaderd mag worden.

Indien een pagina benaderd wordt op een wijze die door het S bit of het WP bit in de Page Descriptor verboden wordt, dan wordt de Bus Error Exception uitgevoerd, sterker nog als bij het afzoeken van de Translation Table ergens een "verkeerd" S of WP bit staat, dan zal de Bus Error Exception uitgevoerd worden onafhankelijk van de bits in de Page Descriptor.

De gebieden die als Unused in de figuur staan, worden door de 68030 niet gebruikt. Hier kan het Operating System eventueel zinvol gebruik van maken door er bijvoorbeeld in te zetten welk block van de schijf hier in het geheugen staat.

De Short Format Page Descriptor kent geen Unused bits en mist ook het S bit. Als deze gegevens niet door het Operating System gebruikt worden, heeft dit formaat uiteraard de voorkeur omdat hiermee

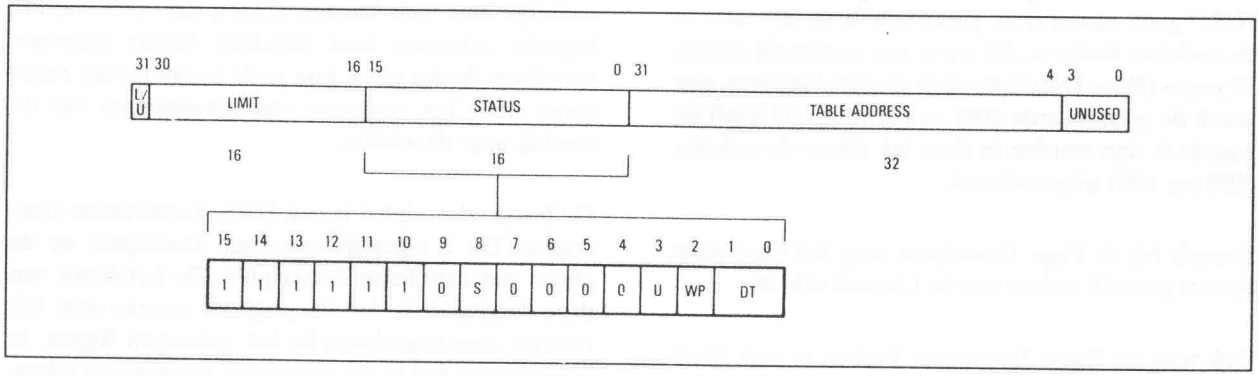


Fig. 6: long format table descriptor

een aanzienlijke besparing in geheugengebruik mee bereikt kan worden.

In figuur 6 is de opbouw van een Table Descriptor weergeven. Ook in dit geval betreft het een Long Format Descriptor. Een aantal velden uit de descriptor zijn al bij de Page Descriptor behandeld (WP, U en S). Het veld DT is weer het Descriptor Type die in dit geval de waarde 10 of 11 heeft waarbij 10 betekent dat de descriptor verwijst naar descriptors met een lengte van 4 byte (Short Format) en 11 dat de descriptor verwijst naar descriptors met een lengte van 8 byte (Long Format). Het Table Adres bevat het adres waarop de tabel met onderliggende descriptors begint. Zoals reeds beschreven is, geeft de inhoud van het Translation Control Register aan hoeveel descriptors een bepaald niveau bevat en welke logische adresbits als index in deze tabel gebruikt worden.

Stel nu dat het Translation Control Register zodanig geprogrammeerd is dat we twee niveaus met descriptors hebben waarbij elk niveau door 4 bits geïndexeerd worden. Op het eerste niveau vinden we in dat geval 16 Table Descriptors. Elke Table Descriptor verwijst vervolgens naar een tabel met daarin 16 Page Descriptors. In totaal verwijzen deze Page Descriptors dus 16 * 16 = 256 pagina's.

Het Table Adres bevat het adres waarop de tabel met onderliggende descriptors begint.

Nu hebben Long Format Table Descriptors nog een extra eigenschap. In het bovengenoemde voorbeeld hebben we, per tabel, te maken met 16 Page Descriptors. In werkelijkheid kunnen dit er veel meer zijn. Stel nu dat we hiervan slechts 10 nodig hebben omdat we gewoon niet meer ruimte nodig hebben. In dat geval kun je met het L/U bit en het Limit veld aangeven dat je niet alle onderliggende takken van de boom wenst te gebruiken. In het veld Limit kan een bitpatroon, = een deel van het logische adres, dat als grenswaarde optreedt. Het veld "L/U" geeft aan of deze grenswaarde de bovenste of de onderste limiet is. Als we in ons voorbeeld slechts 10 pages (Page Descriptors) in de tabel hebben, dan wordt de grenswaarde 1001 en het veld L/U heeft de waarde 0, dan worden in deze tak alleen de indexen 0000 t/m 1001 uitgecodeerd.

Evenals bij de Page Descriptor mag het Operating System gebruik maken van de Unused velden.

Ook voor de Table Descriptor bestaat er een Short formaat. Hier vervallen, naast de Unused velden, het S bit en de Limit mogelijkheid.

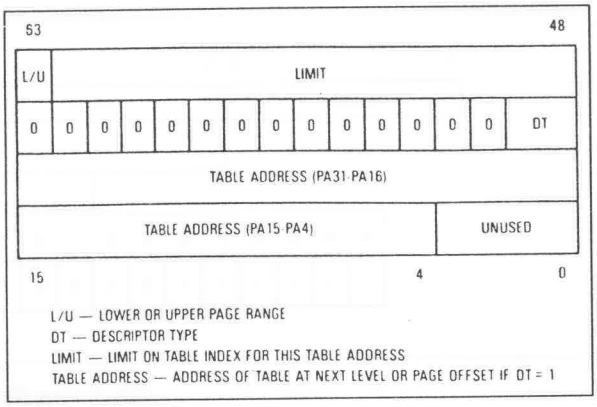


Fig. 7: root pointer descriptor

De root van de vertaaltabel staat in de Root Pointer Descriptor. Als de vertaaltabel actief is, dan wordt deze descriptor opgeslagen in één van de twee Root Pointer registers van de MMU. Het formaat van deze descriptor is afgeleid van de Long Format Table Descriptor en staat weergegeven in figuur 7.

Behalve de hier behandelde descriptors kent de MMU nog enkele descriptors die niet zo uitgebreid behandeld zullen worden. In de eerste plaats is dit de zogenaamde Invalid Descriptor (DT = 00). Hiermee kan worden aangegeven dat een tak van de vertaaltabel niet ingevuld is. Dit kan bijvoorbeeld betekenen dat het Operating System het (nog) niet nodig vond deze tak in te vullen of dat de tak naar schijf is weggeschreven om fysiek geheugen uit te sparen. De 68030 zal een Bus Error Exception uitvoeren als de MMU een dergelijke descriptor tegenkomt.

Een andere descriptor is de Indirect Descriptor. Hiermee kan in een tak een verwijzing naar een andere tak opgenomen worden om daarmee geheugenruimte voor de vertaaltabel uit te sparen. Het zou namelijk best voor kunnen komen dat verschillende logische adressen naar dezelfde fysieke adressen verwijzen. In dat geval kun je de boom aardig reduceren door het opnemen van verwijzingen van de ene tak naar de andere.

De laatste descriptor is een Early Termination Descriptor. Dit is eigenlijk een Page Descriptor op de plaats van een Table Descriptor. De betekenis van deze descriptor is dat alle pagina's waarna deze tak verwijst aanééngesloten in het geheugen liggen, te beginnen op het in de descriptor aangegeven adres. Ook hiermee kun je een vertaaltabel flink reduce-

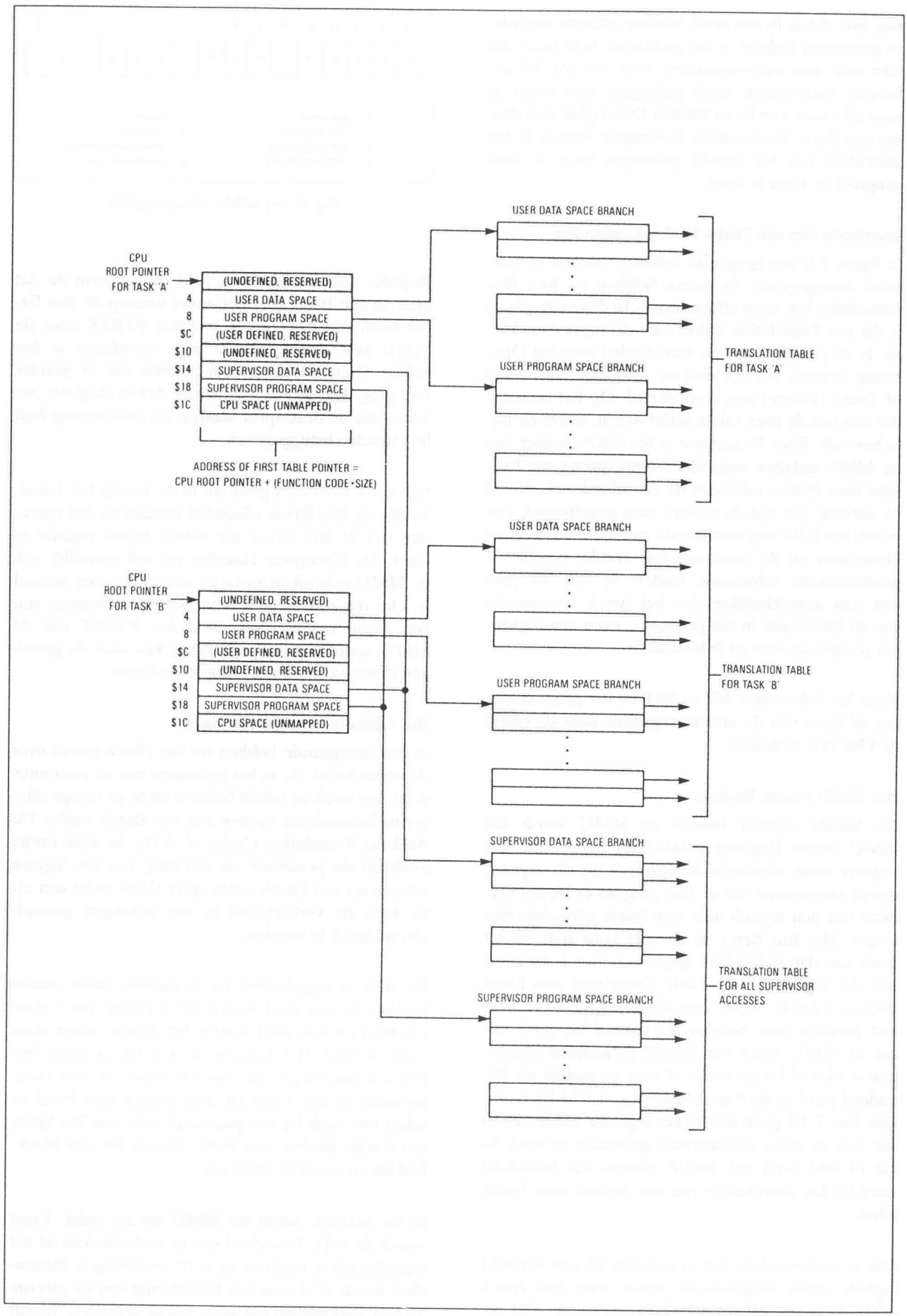


Fig. 8: vertaaltabel in een multi-tasking omgeving

ren. Stel dat je in een multi-tasking systeem meerdere processen (taken) in het geheugen hebt maar dat elke taak een aaneengesloten blok van 256 kB geheugen toegewezen heeft gekregen, dan maak je voor elke taak een Root Pointer Descriptor met daarin een Early Termination Descriptor waarin je het startadres van het fysieke geheugen voor de taak aangeeft en klaar is Kees.

Voorbeeld van een Multi-Tasking omgeving

In figuur 8 is een mogelijke opbouw van een vertaaltabel weergegeven. In wezen hebben we hier drie vertaaltabellen door elkaar lopen. In de eerste plaats is dat een Translation Table voor de Supervisor Mode. In de praktijk is dit de vertaaltabel voor het Operating System. Verder zien we voor twee processen of Tasks (Taken) een vertaaltabel. Op het moment dat één van de twee taken actief wordt, wordt de bijbehorende Root Descriptor is het CRP-register van de MMU geladen waarna de omrekening van logische naar fysieke adressen via die tabel loopt. Wordt na verloop van tijd de andere taak geactiveerd, dan wordt het CRP-register gevuld met de nieuwe Root Descriptor en de processor kan zonder moeite de geadresseerde informatie vinden. Je ziet, dat gaat een stuk gemakkelijker dan het fysiek verschuiven van de informatie in het geheugen, even een register om programmeren en je bent klaar.

Voor het laden van CRP en SRP en het programmeren of lezen van de andere registers, kent de 68030 de PMOVE instructie.

Het MMU Status Register

Als laatste register binnen de MMU wordt het MMU Status Register (MMUSR) behandeld. Dit register staat afgebeeld in figuur 9. In dit register wordt aangegeven wat er fout ging als er bij het vertalen van een logisch naar een fysiek adres iets niet klopte. Het Bus Error bit spreekt voor zich, dit bit geeft aan dat er iets fout gegaan is. Het L bit geeft aan dat ergens in een Table Descriptor een Limit overschreden is. Supervisor Violation en Write Protect spreken voor zich en het Invalid bit geeft aan dat de MMU tegen een Invalid Descriptor opgelopen is. Het M bit geeft aan of voor de pagina die benaderd werd in de Page Descriptor het M bit 0 of 1 was. Het T bit geeft aan of het logische adres zich in één van de twee transparante gebieden bevond. In het N veld staat het aantal niveaus dat benaderd werd bij het omrekenen van een logisch naar fysiek adres.

Om te onderzoeken wat er gebeurt als een bepaald logisch adres omgerekend wordt naar een fysiek adres, kent de 68030 de PTEST instructie. Het resultaat van de omrekening wordt in het MMU Status

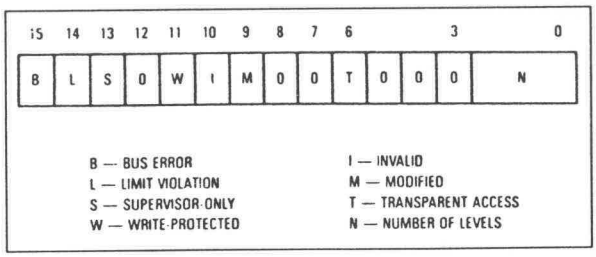


Fig. 9: het MMU status register

Register geschreven. In de praktijk betekent dit dat men na een Bus Error het adres waarop de Bus Error heeft plaatsgevonden middels PTEST door de MMU laat testen waarna men vervolgens in het MMU Status Register kan aflezen wat er precies fout ging. Bovendien kan in een Adres Register het adres van de descriptor waarop de omrekening fout liep worden teruggegeven.

Het heeft uiteraard geen zin in de Exception Handler die de Bus Error afhandelt meteen na het optreden van de Bus Error het MMU Status register te lezen. De Exception Handler zal zelf namelijk ook de MMU gebruiken zodat de informatie niet actueel is. Om die reden moet men een adres waarop een Bus Error optrad opnieuw m.b.v. PTEST aan de MMU aanbieden en pas daarna kan men de gevolgen in het MMU Status Register aflezen.

Het Address Translation Cache

In het voorgaande hebben we het alleen gehad over de vertaaltabel die in het geheugen van de computer staat. Nu heeft de 68030 behalve de in de vorige aflevering behandelde caches nog een derde cache: De Address Translation Cache of ATC. In deze cache probeert de processor de vertaling van een logisch adres naar een fysiek adres op te slaan zodat niet elke keer de vertaaltabel in het geheugen geraadpleegd hoeft te worden.

De ATC is opgebouwd uit 22 entries. Deze entries bestaan uit een deel waarin het logische adres staat (28 bits) en een deel waarin het fysieke adres staat (ook 28 bits). Het logische deel bevat 24 adres bits (bij een pagina-grootte van 256 byte), de drie functiecodes en een Valid bit. Het fysieke deel bevat 24 adres bits (ook bij een pagina-grootte van 256 byte), een Cache Inhibit, een Write Protect bit, een Modified bit en een Bus Error bit.

In de praktijk werkt de MMU nu als volgt: Eerst wordt de ATC benaderd om te onderzoeken of het logische adres reeds in de ATC aanwezig is. Bovendien wordt er al vast een benadering van de externe vertaaltabel gestart om geen tijd te verliezen. Wordt het logische adres in de ATC gevonden, dan wordt

de externe benadering afgebroken. Staat het logische adres niet in de ATC, dan zal de vertaaltabel afgezocht worden en wordt er een entry in de ATC met de omrekening gevuld. Vervolgens wordt aan de hand van de ATC het nu bekende fysieke adres benaderd.

Wordt er bij het benaderen van de externe tabel een fout geconstateerd, dan wordt er niet meteen een Bus Error uitgevoerd. Eerst wordt de entry in de ATC geladen en wordt het Bus Error bit in de ATC gezet. Als vervolgens de ATC benaderd wordt om het fysieke adres te bepalen, zal de Bus Error Exception optreden. Alle benaderingen van het fysieke geheugen gaan dus altijd via de ATC.

Als er een pagina benaderd wordt voor een schrijfoperatie en het M-bit in de ATC-entry is clear, dan zal ook eerst de externe tabel geraadpleegd worden om het M-bit in de Page Descriptor (en de ATC-entry) goed te zetten; pas daarna zal de omrekening van logisch naar fysiek adres overgenomen worden.

Wat gebeurt er nu als er een (onbekende) omrekening gevraagd wordt en de ATC is vol? Op dat moment zal de MMU proberen de entry die het langst niet gebruikt is te wissen waarna hier de nieuwe informatie in wordt geschreven. Op deze manier worden in de praktijk tot ruim 98% van de omrekeningen in de ATC gevonden; al weer een bewijs van de kracht van de 68030 processor. Bovendien kent de 68030 instructies waarmee entries expliciet in de ATC opgenomen kunnen worden of uit de ATC kunnen worden gehaald.

Afsluiting

Zo, dat was de MMU. Ik ben zelf de laatste dagen (bij het schrijven van dit artikel) behoorlijk onder de indruk geraakt van de mogelijkheden die de 68030 MMU heeft. Het koste me wel enige studie om alles enigszins te begrijpen maar zoals ik het nu zie is het toch redelijk eenvoudig te gebruiken. Nu alleen nog een Operating System die hier optimaal gebruik van maakt en we hebben inderdaad de krachtige KGN68k computer die ons bij de start van het project voor ogen stond.

Over de processor zelf ben ik nu wel zo'n beetje uitgeschreven. Betekent dit nu het einde van deze serie? Ik hoop het niet! Door een aantal omstandigheden (o.a. het wegvallen van de redacteur) ben ik zelf al enige maanden niet meer actief binnen de werkgroep maar ik hoop nu weer wat dingen op te kunnen pakken. Dat zal zeker betekenen dat er behalve over de processor ook een hoop kennis over de gebruikte periferie chips vergaard zal moeten worden. Kennis die ook weer via artikelen aan de leden overgedragen kan worden. Verder zal er nu toch ook echt een goede start gemaakt moeten worden met een Operating System waarbij er vast wel weer nieuwe stof tot schrijven bovenkomt en tenslotte hebben de leden nog altijd een beschrijving van de ontwikkelde hardware te goed.

Kortom, de serie zal onder de huidige naam stoppen maar tegelijkertijd zal er een nieuwe serie starten over andere onderwerpen die met KGN68k te maken hebben. Tot de volgende keer dus.

Literatuur

1: Motorola: MC68030 Enhanced 32-bit microprocessor user's manual; Prentice Hall 1990. ISBN: 0-13-566423-3.
2: Werner Hilf: Der 68030; MC 4, 5 en 6 1988, Franzis Verlag GmbH.

Documentatie 1 is beschikbaar gesteld aan de KGN68k werkgroep door EBV Elektronik te Maarssenbroek.

Gert van Opbroek

Voortgang DOS65

Gisteren (17 april) is de werkgroep DOS65 weer bijeen geweest. Er is weer gebrainstormd over hoe het verder moet. De sfeer was zoals gebruikelijk goed, en de zin om verder te gaan ontbrak evenmin.

De belangrijkste ontwikkeling van de laatste maand is dat er een dubbelzijdige print is getekend van de 1 Mbyte RAMkaart uit het vorige nummer. De print staat op deze bladzijden afgebeeld. Erwin Visschedijk heeft ondergetekende enige tekenvaardigheden bijgebracht, en samen hebben we de print getekend. Speurders die het vorige nummer erbij pakken zullen een paar verschillen ontdekken: de nummering van de IC's is veranderd en er is een batterijtje met bijbehorende chip verschenen. De kaart is nu namelijk ook voorzien van battery-backup, met behulp van de Dallas DS1210 "Nonvolatile controller".

Wat absoluut niet is gelukt is de print enkelzijdig te houden. Dat werd al duidelijk toen we het tekenpakket hadden verteld hoe groot de print moest worden (Eurokaart) en de netlist was ingevoerd: het leek er bijna op dat de print overvol zou worden. Dat bleek mee te vallen, maar om te spreken van een totaal lege print is anders.

De volgende stap is het verder debuggen van het prototype, en het bestellen van een aantal proefprinten. Als deze werken kunnen we aan de slag met DOS65 V3.00. De werkgroep heeft inmiddels de volgende taakverdeling afgesproken:

Tonny Schäffer:	Contact leggen met Ad Brouwer.
Jaap Prenger:	Napeinzen over de combokaart, een kaart met een SCSI harddisk interface, een real time clock, reset- en NMI-generators en misschien wat extra RS-232 poort(en).
Antoine Megens:	IO65 aanpassen.
Henk Speksnijder:	IO65 aanpassen.
Nico de Vries:	1 Mbyte RAMkaart voltooien.

Kortom, er is werk genoeg aan de winkel. Binnenkort hoort u wat het kostenplaatje voor de RAMkaart gaat worden, en hoe u aan de print, de PAL en eventuele andere moeilijke onderdelen kunt komen. Tot de volgende keer.

Nico de Vries

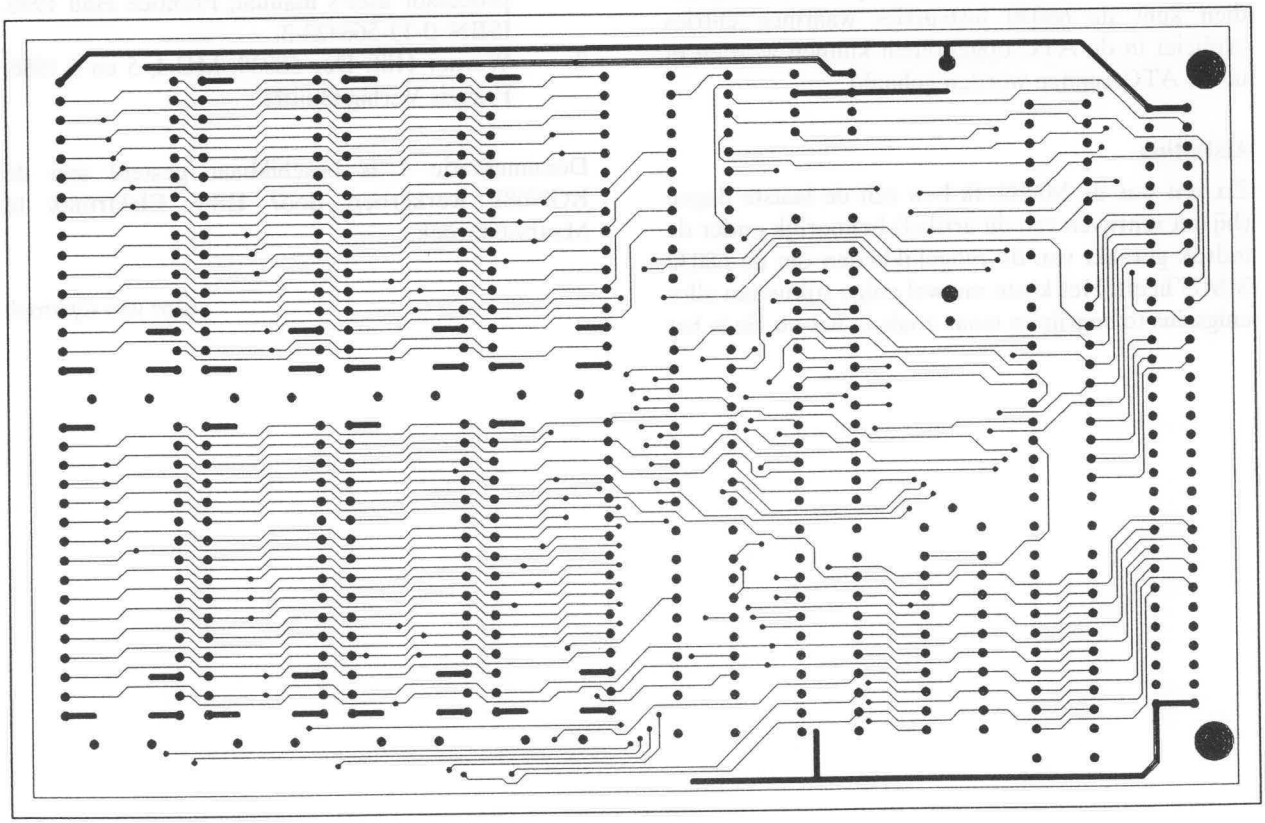


Fig. 1: print, sporen aan soldeerzijde

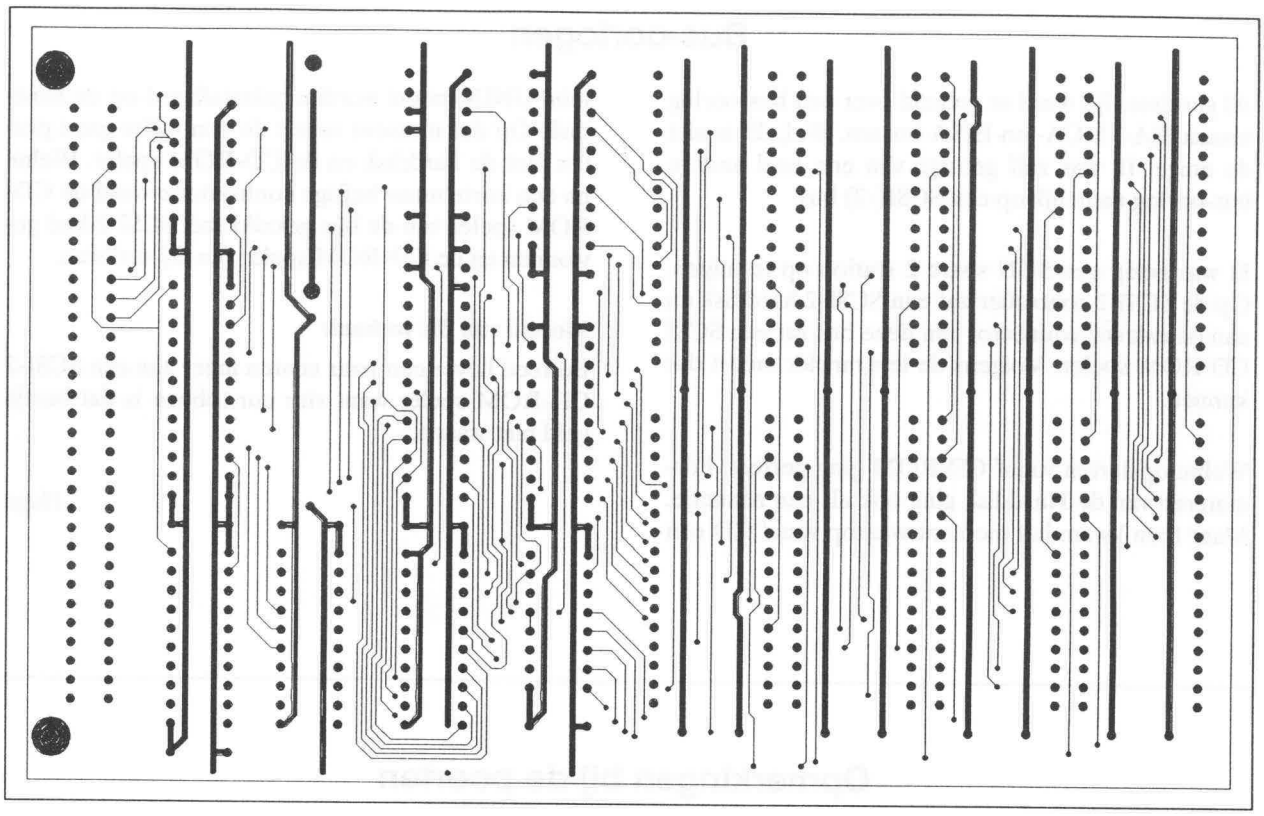


Fig 2: print, sporen aan componentenzijde

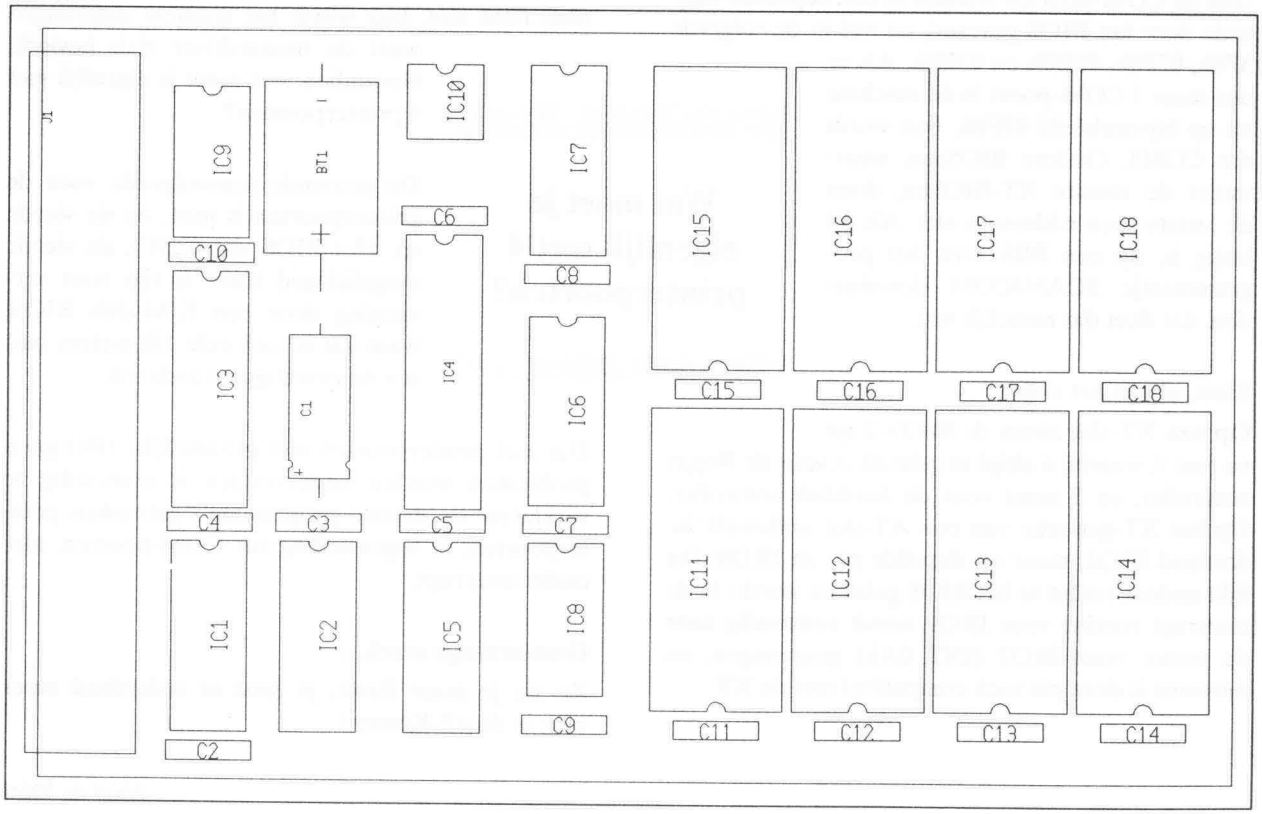


Fig 3: print, componentenopstelling

Bus-oorlogen

Al geruime tijd werd er gemeld over een bus-oorlog tussen ISA-, MCA- en EISA-bussen. Wel, dit is niet de enige. Ik was zelf getuige van een heel andere bus-oorlog namelijk op een SCSI(-2) bus.

Ik was bezig een SUN sparc 2 station op te tuigen. Op de SCSI-2 controller zat een SCSI-2 harddisk en aan de externe connector van deze bus zat een SCSI CD-ROM speler. Volgens de leverancier moest dat kunnen.

Welnu opstarten vanaf CD-ROM ging perfect. Formateren van de Harddisk ging ook als een zonnetje. Maar toen kwam het moment waarop vanaf CD een mini UNIX moest worden geïnstalleerd op de harddisk. Op dat moment moest de controller gaan praten met de harddisk en de CD-ROM speler. Welnu na een korte maar heftige confrontatie werd de CD-ROM speler van de bus geknikkerd. SCSI-2 had gewonnen en de CD-ROM speler (en ik) verloren.

Moraal van dit verhaal:

Besteed liever een paar centen meer aan een SCSI-2 CD-ROM speler want vier uur tobben is dat beetje geld niet waard.

Hugo

Opmerkingen bij de poorten

In het artikel "Poorten in een PC" uit het vorige nummer zitten helaas een paar onjuistheden. Als eerste de COM-poorten.

COM-poorten

Ook de COM-poorten worden in een bepaalde volgorde door het BIOS gescand, en wel in de volgorde 3F8h, 02F8h, 03E8h en 02E8h. Als er dus maar 1 COM-poort in de machine zit op bijvoorbeeld 02F8h, dan wordt dat COM1. Oudere BIOSsen, waaronder de meeste XT-BIOSen, doen de laatste twee addressen niet. Als dit lastig is, op een BBS even het programmaatje SCAN4COM downloaden, dat doet dat namelijk wel.

Slots, of zijn het sloten?

Op een XT-slot zitten de IRQ's 2 tot en met 7, waarbij 6 altijd in gebruik is voor de floppy controller, en 5 soms voor de harddisk controller. Op het XT-gedeelte van een AT-slot ontbreekt inderdaad IRQ2, maar op dezelfde pin zit IRQ9. Dit lijkt anders, totdat in het BIOS gekeken wordt: In de interrupt routine voor IRQ9 wordt eenvoudig naar de vector voor IRQ2 (INT 0Ah) gesprongen, en daarmee is deze pin toch compatibel met de XT.

Paardelul (eh... pardon, parallel)poorten

Op alle PC's is het helaas niet mogelijk 4 parallelle poorten aan te sluiten, omdat er in het BIOS maar ruimte is voor drie LPT poortadressen. Het vierde adres (0040:000E) lijkt wel beschikbaar te zijn, maar is dit in een aantal BIOSsen, waaronder die van Big Blue IBM niet: hier wordt het segment opgeslagen waar de mousedriver zich bevindt. Bovendien: wat moet je eigenlijk met 4 printerpoorten?

Wat moet je eigenlijk met 4 printerpoorten?

De getoonde scanvolgorde voor de printerpoorten is juist, op de vierde na. Het BIOS dat 02BCh als vierde mogelijkheid scant is rijp voor vervanging door een KIM-club BIOS, want dat is toch echt kilometers buiten de gevestigde standaard...

Dat met printerpoorten met gezamelijke IRQ geen problemen worden ondervonden is eenvoudig te verklaren: de meeste programma's gebruiken printerpoorten, in tegenstelling tot COM-poorten, niet onder interrupt.

Geen ernstige seack...

Zo zie je maar Ernst, je leest er inderdaad meer over in de μP-Kenner!

Nico de Vries

Van de bestuurstafel

Door paniek bij mijn vriendin over een centrifuge die in de brand stond en mijn vlucht naar haar huis was ik bijna vergeten dat er nog iets voor ons lijfblad geschreven moest worden. Als het dus over komt dat dit stukje in de haast is gemaakt dan klopt dat wel. Maar het leven van een voorzitter gaat nu eenmaal niet altijd over rozen.

Eén van die dorens is de bezetting van het bestuur. We zijn momenteel onderbezet zoals jullie weten. Binnen het bestuur is dat goed te merken soms raakt één van ons een beetje oververhit omdat hij teveel moet doen. Dat is niet zo verwonderlijk, uiteindelijk hebben we ook nog een privéleven. Het is echt niet zo dat het besturen zoveel tijd vraagt maar als je het werk van zeven man met z'n vieren moet gaan doen vergt dat behoorlijk meer tijd dan de bedoeling is. We, en daar bedoel ik jullie ook mee, moeten oppassen dat door het oververhit raken er niet nog meer bestuursleden de pijp aan Maarten geven want dan ontstaat er een onbestuurlijke vereniging. Dat is niet onze bedoeling, maar we staan er niet voor in dat dat toch gebeurt. Het is aan de leden om voor versterking te zorgen, het is ook jullie vereniging.

Zoals jullie kunnen zien is dit blad weer goed gevuld. Dit hebben we te danken aan een paar leden en het bestuur. Denk nu niet het gaat wel goed, we kunnen blijven zitten. Dat is beslist niet zo. We kunnen altijd kopij gebruiken. Denk eens aan een ingezonden brief met opbouwende kritiek. Of een artikeltje van een of een halve pagina. Daar zitten we nog steeds om te springen. Ons lijfblad is er toch voor en door de leden. Dus klim eens in de pen en laat van je horen.

Nu niet meteen denken dat ik alleen wat te zeuren heb. Voor de DOS65 mensen heb ik goed nieuws. De werkgroep heeft besloten om DOS65 nieuw leven in te blazen. Op de bijeenkomst in Geldrop heeft er een levendige discussie plaats gevonden over wat er zou moeten gebeuren om DOS65 te laten herleven. Ideeën van de werkgroep werden positief ontvangen. De ideeën waren o.a. een nieuwe geheugen kaart waarop minimaal 1M statische RAM komt. En een harddiskcontroller. Omdat op de laatste kaart waarschijnlijk nog ruimte over is wordt er aan gedacht om daar een RTC, een RS-232 interface, een SCSI-interface en zo op te plaatsen. Al met al een leuke uitbreiding en voor de toekomst en weer soldeerplezier voor de DOS65 bezitters. Ja: jullie lezen het goed, de werkgroep heeft inmiddels besloten om door te gaan. We hopen dat er op een volgende DOS65 bijeenkomst wel meer animo is dan in Geldrop. Maar met een aantal interessante uitbreidingen moet dat toch wel lukken. Laat de werkgroep zien dat ze er voor jullie zijn. We zullen tijdig bekend maken wanneer we weer zo'n speciale DOS65 bijeenkomst gaan plannen.

De komende bijeenkomst in Utrecht staat in het teken van TeX: een Public Domain documentopmaakprogramma waar Phons Bloemen het één en ander over komt vertellen. Het programma zal daarna ook beschikbaar zijn op ons bulletin-bord. Utrecht ligt centraal dus we verwachten dat ook jij je gezicht daar eens laat zien: bijeenkomsten horen bij het verenigingsleven en niet alleen het lezen van ons clubblad. Tot ziens dus op 15 mei in Utrecht.

Uw voorkauwer

Nieuwtjes en andere wetenswaardigheden

Komt er een Apple-kloon?

Het zou best wel eens kunnen. Bij Apple zit een bedrijfje om de hoek, NuTek geheten. Vorig jaar lieten ze al een computer zien die met een Motiv-interface Macintosh-programma's draaide. Het bedrijf levert nu machines met een eigen besturingssysteem dat de eigenschappen van de Macintosh nabootst. Die machine is nog niet helemaal compleet. Maar ze beweren dadelijk een complete machine met 33MHz 68030-processor te kunnen leveren voor een prijs van ongeveer 1600 dollar.

Nog iets over appels

Onder de naam Apple Macintosh Hardware System Update Version 1.0 stelt Apple voor de Macintosh LC, LC II, IIsi, IIvx, IIvi, Classic II, Quadra 900 en 950 een upgrade beschikbaar. Deze systeemverbetering voor System 7.1 verbetert de klok, de datacommunicatie en voorkomt problemen met de schijfeenheid. Deze verbetering is in principe kosteloos te verkrijgen bij de geautoriseerde Apple verkopers.

Hier spreekt Lantastic

Lantastic brengt een sterk verbeterde versie 5 uit. Buiten alle andere verbeteringen biedt versie 5 gebruikers van Windows voice chat en gesproken foutmeldingen. Hiervoor is wel het Artisoft Sounding Board nodig. Maar dan heb je ook een sprekend netwerk.

Informatie

De μP Kenner (De microprocessor Kenner) is een uitgave van de KIM gebruikersclub Nederland. Deze vereniging is volledig onafhankelijk, is statutair opgericht op 22 juni 1978 en ingeschreven bij de Kamer van Koophandel en Fabrieken voor Hollands Noorderkwartier te Alkmaar, onder nummer 634305. Het gironummer van de vereniging is 3757649.

De doelstellingen van de vereniging zijn sinds 1 januari 1989 als volgt geformuleerd:

- Het vergaren en verspreiden van kennis over componenten van microcomputers, de microcomputers zelf en de bijbehorende systeemsoftware.
- Het stimuleren en ondersteunen van het gebruik van micro-computers in de meer technische toepassingen.

Om deze doelstellingen zo goed mogelijk in te vullen, wordt onder andere 5 maal per jaar de μP Kenner uitgegeven. Verder worden er door het bestuur per jaar 5 landelijke bijeenkomsten georganiseerd, beheert het bestuur een Bulletin Board en wordt er een softwarebibliotheek en een technisch forum voor de diverse systemen in stand gehouden.

Landelijke bijeenkomsten:

Deze worden gehouden op bij voorkeur de derde zaterdag van de maanden januari, maart, mei, september en november. De exacte plaats en datum worden steeds in de μP Kenner bekend gemaakt in de rubriek Uitnodiging.

Bulletin Board:

Voor het uitwisselen van mededelingen, het stellen en beantwoorden van vragen en de verspreiding van software wordt er door de vereniging een Bulletin Board beschikbaar gesteld.
De telefoonnummers zijn: 053-328506, 053-303902 of 053-327457.

Software Bibliotheek en Technisch Forum:

Voor het beheer van de Software Bibliotheek en technischeondersteuning streeft het bestuur ernaar zgn. systeemcoordinators te benoemen. Van tijd tot tijd zal in de μP Kenner een overzicht gepubliceerd worden. Dit overzicht staat ook op het Bulletin Board.

Correspondentie adres

Alle correspondentie betreffende verenigingszaken kan gestuurd worden aan:

KIM Gebruikersclub Nederland
Postbus 1336
7500 BH Enschede

Het Bestuur

Het bestuur van de vereniging wordt gevormd door een dagelijks bestuur bestaande uit een voorzitter, een secretaris en een penningmeester en een viertal gewone leden.

Tonny Schäffer (voorzitter)
Paul Krügerstraat 27
7532 PW Enschede
Telefoon 053-613678

Jacques H.G.M. Banser (penningmeester)
Haaksbergerstraat 199
7513 EM Enschede
Telefoon 053-324137

Gert van Opbroek (secretaris/redactie μP Kenner)
Bateweg 60
2481 AN Woubrugge
Telefoon 01729-8636

Geert Stappers (KGN/68k coordinator)
Engelseweg 7
5825 BT Overloon
Telefoon 04781-41279

Ereleden:

Naast het bestuur zijn er een aantal ereleden, die zich in het verleden bijzonder verdienstelijk voor de club hebben gemaakt:

Erevoorzitter:
Siep de Vries

Ereleden:
Mevr. H. de Vries-van der Winden
Anton Müller
Rinus Vleesch Dubois